IBM Data Language/I Virtual Storage Extended 1.12.1

Release Guide

SC33-6211-09

Note!
Before using this information and the product it supports, be sure to read the general information under “Notices” on page 117.

This edition applies to Version 1 Release 12.1 of IBM Data Language/I Virtual Storage Extended (DL/I VSE), Program Number (5746-XX1).

Order publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the addresses given below.

A form for readers' comments is provided at the back of this publication. If the form has been removed, address your comments to:

IBM Deutschland Research & Development GmbH
Department 3282
Schoenaicher Strasse 220
D-71032 Boeblingen
Federal Republic of Germany

You may also send your comments by FAX or via the Internet:

Internet: s390id@de.ibm.com
FAX (Germany): 07031-16-3456
FAX (other countries): (+49)+7031-16-3456

When you send information to IBM, you grant IBM a non-exclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Contents

Figures vii

Tables ix

About This Publication xi
Who Should Use This Publicaton xi
How to Use This Publication xi
Where to Find More Information xi

Summary of Changes for DL/I xiii

Chapter 1. Release News for DL/I VSE 1.12.1 1
Features Introduced with DL/I VSE 1.12.1 1
DL/I Partitioning 1
Enhancements introduced through APAR fixes in DL/I VSE 1.12.1 5
DL/I VSE 1.12.0 Enhancements 7
Migrating to DL/I VSE 1.12.1 7
New and Changed Messages 7

Chapter 2. Operating Environment 9
Machine Requirements 9
Running DL/I in a VM Environment 9
Programming Requirements 10
Programming Environment 10

Chapter 3. Installation 11
DL/I Shipment Details 11
Production Sublibrary 11
Generation Sublibrary 14

Chapter 4. Migration and Compatibility Considerations 15
Migration from DL/I 1.7.0 to 1.7.1 15
Migration from DL/I 1.7.1 to 1.8.0 15
Migration from DL/I 1.8.0 to 1.9.0 16
Migration from DL/I 1.9.0 to 1.10.0 16
Migration from DL/I 1.10.0 to 1.11.0 16
Migration from DL/I 1.11.0 to 1.12.0 16
Migration from DL/I 1.12.0 to 1.12.1 17
Compatibility with IMS/VS 17

Chapter 5. Customizing DL/I 19
SVA Loading for DL/I 19
CICS Adaptation 20
DL/I IMF and IUG Functions 21
DL/I Documentation Aid and the ISQL EXTRACT DEFINEs Utility 22
DL/I Documentation Aid 23
ISQL EXTRACT DEFINEs 24

Chapter 6. Post Installation Tasks and Tools....27

Chapter 7. Applying Service....31
Error Messages while Reassembling DL/I Components....31
Error Messages while Relinking DL/I Components....32

Chapter 8. Features Introduced with Earlier Releases of DL/I....35
Features Introduced with DL/I DOS/VS 1.7.1....35
Improved Multiple Partition Support....35
Dynamically Scheduling MPS or Non-MPS Execution....35
Features Introduced with DL/I DOS/VS 1.8....37
Date and Time on Reports and Statistics....37
Literal String in the HLPI WHERE Clause....37
Features Introduced with DL/I DOS/VS 1.9....39
Dynamic Partition Support....39
Database I/O Error Handling....40
DL/I in a CICS/XRF Environment....40
Features Introduced with DL/I DOS/VS 1.10....42
DL/I Applications above the 16 MB Line of Storage....42
VSE/VSAM HS-Buffers Above the 16 MB Line of Storage....42
Virtual Disk Exploitation....44
DL/I Run Statistics (DLZSTTL) Enhancements....44
Features Introduced with DL/I VSE 1.11....46
Multiple MPS Systems....47
More than 4 GB for HD Databases....48
Support of CICS Transaction Server Storage Protection....51
Conditional SVA Loading....52
DL/I DOS/VS 1.10 Enhancements....52
Migrating to DL/I VSE 1.11 and the CICS Transaction Server for VSE/ESA 1.1....56
DL/I VSE 1.11 Migration Items....56
CICS Transaction Server for VSE/ESA 1.1 Migration Items....57
CICS Coexistence Environment....58
Features Introduced with DL/I VSE 1.12.0....58
Allocation of DL/I Resources above 16 MB....58
DL/I GETVIS Storage in Separate Subpools....60
Conditional SVA Loading Disabled....61
DL/I VSE 1.11 Enhancements....61
Migrating to DL/I VSE 1.12.0....65
Migration Items from New DL/I VSE 1.12.0 Functions....65
Migration Items from DL/I VSE 1.11 APARs....67

Chapter 9. Recovery after a DLZ004I or DLZ005I Error....69
DLZ004I / DLZ005I in a CICS / DL/I Online Environment....69
DLIOER=ABEND....70
DLIOER=CONTINUE....71
Operating in a CICS XRF Environment....72
Recovering from missing or inappropriate DL/I Recovery....72
DLZ004I / DLZ005I in a DL/I Batch Environment....73
DL/I Batch Program Terminated Abnormally....73
DL/I Batch Program Continued Execution....73
Recovering from missing or inappropriate DL/I Recovery....73
DL/I Batch Recovery Processes....74
DL/I Batch Recovery Process I....74
DL/I Batch Recovery Process II....76
DL/I Batch Recovery Process III....79
DL/I Batch Recovery Process IV....82

DL/I Batch Recovery Process V....83
DL/I Batch Recovery Process VI....86
DL/I Batch Recovery Process VII....88
Start a new Log Cycle and Take new Backups....91
Restart CICS and DL/I after DL/I Batch Recovery....91
Responses to Message DLZ142A....92

Chapter 10. Messages and Codes....93
New and Changed Messages....93
New DL/I CICS Return Codes....106

Appendix A. Sample Job Streams for DL/I 31-Bit Applications....107
DL/I - Online Applications....107
DL/I - Batch Applications....109

Appendix B. Summary of Customer Interfaces....113
DL/I Macros Intended for Customer Use....113

Related IBM Publications....115

Notices....117
Programming Interface Information....118
Trademarks....118
Terms and Conditions for Product Documentation....118

Accessibility....121
Using Assistive Technologies....121
Documentation Format....121
Glossary....123

Index....127

Figures

1. Partitioning a database..... 1

2. Sample Job SVALOAD1..... 19

3. Sample Job SVALOAD2..... 20

4. Sample Job DELETVM..... 22

5. Sample Job PUNCHDA..... 23

6. Sample Job PREP DLZDLBD..... 25

7. Sample Job DSPLYPRD..... 27

8. Sample Job DSPLYGEN..... 28

9. How to Invoke the New GSTA Call in an Assembler Program..... 36

10. Layout of the Statistics Buffer DLZSTBF..... 37

11. Example: Output of the DL/I Run Statistics Program (DLZSTTL)..... 45

12. ...continued Example: Output of the DL/I Run Statistics Program (DLZSTTL)..... 46

13. Examples of the PARTID and APPLID Parameters..... 47

14. DBD Definition with Multiple DATASET Statements..... 50

15. ....continued DBD Definition with Multiple DATASET Statements..... 51

16. Examples of Valid 'PARM=' strings..... 62

17. COBOL Online with HLPI or CALL Interface..... 107

18. PL/I Online with HLPI or CALL Interface..... 108

19. Example: High Level Assembler - Online..... 108

20. COBOL Batch and Batch/MPS with HLPI Interface..... 109

21. COBOL Batch and Batch/MPS with CALL Interface..... 109

22. PL/I for VSE/ESA Batch and Batch/MPS with HLPI Interface..... 110

23. PL/I for VSE/ESA Batch and Batch/MPS with CALL Interface..... 110

24. Example: High Level Assembler - Batch and MPS Batch......111

Tables

1. Partitioning a database description (DBD)..... 2

2. Database partitioning - segment lengths..... 3

3. Naming Convention for DL/I DA and ISQL EXTRACT Modules..... 24

4. GSTA Call Return Codes..... 36

5. Relational Operators of a DL/I SSA.....64

6. Summary of Customer Interfaces..... 113

About This Publication

This publication describes the latest enhancements provided with DL/I VSE 1.12.1 available since z/VSE 6.2.

It also includes information on installation, migration, customization, and other post-installation tasks. Changed and new messages are included as well.

The publication includes information on earlier DL/I releases and environments for reference purposes.

Who Should Use This Publicaton

This publication is for people who are responsible for setting up a DL/I system in a z/VSE environment.

The assumption is that you have experience with the z/VSE environment in which you want to install DL/I, and that you are familiar with the concepts and terminology of DL/I.

How to Use This Publication

Read first Chapter 1, “Release News for DL/I VSE 1.12.1,” on page 1 to become familiar with the latest enhancements and functions provided by DL/I VSE 1.12.1. The section also includes migration information.

For an overview on features introduced with earlier releases of DL/I, refer to Chapter 8, “Features Introduced with Earlier Releases of DL/I,” on page 35.

Where to Find More Information

For a list of all DL/I publications available, refer to “Related IBM Publications” on page 115.

z/VSE IBM Documentation

IBM Documentation is the new home for IBM's technical information. The z/VSE IBM Documentation can be found here:

https://www.ibm.com/docs/en/zvse/6.2

You can also find VSE user examples (in zipped format) at

https://public.dhe.ibm.com/eserver/zseries/zos/vse/pdf3/zVSE_Samples.pdf

Summary of Changes for DL/I

The following summarizes the changes and enhancements provided with the various DL/I releases.

DL/I VSE 1.12.1 Enhancements

- DL/I Partitioning
- Enhancements introduced through APAR fixes in DL/I VSE 1.12.1
- Enhancements introduced through APAR fixes in DL/I VSE 1.12.0

DL/I VSE 1.12.0 Enhancements

- Allocation of DL/I Resources above 16 MB
- DL/I GETVIS Storage in Separate Subpools
- Conditional SVA Loading Disabled
- Enhancements introduced through APAR fixes in DL/I VSE 1.11

DL/I VSE 1.11 Enhancements

- Multiple MPS Systems
- More than 4 GB for HD Databases
- Support for
  - CICS Transaction Server Storage Protection
  - Conditional SVA Loading
- Enhancements introduced through APAR fixes in DL/I DOS/VS 1.10

DL/I DOS/VS 1.10 Enhancements

- Support for VSE/ESA:
  - DL/I applications above the 16MB line of storage
  - VSE/VSAM HS-buffers above the 16MB line of storage
  - Virtual disk exploitation
- Other changes:
  - Improved DL/I run statistics (APAR PN21734)
  - Improved handling of database I/O errors (APAR PN09116)

DL/I DOS/VS 1.9 Enhancements

- Support of VSE/ESA:
  - CICS/XRF (Extended Recovery Facility)
  - Dynamic Partitions
- Device Independence
- Automatic Verification for DL/I ESDS Data Sets
- Program Isolation Enhancement

DL/I DOS/VS 1.8 Enhancements

- Device Independence (APAR PL48345)
- Automatic Verification for DL/I ESDS Data Sets (APAR PL20988)
- Program Isolation Enhancement (APAR PL55587)

Chapter 1. Release News for DL/I VSE 1.12.1

Note:

DL/I VSE 1.12.1 requires z/VSE 6.2. It cannot be used on z/VSE 6.1 or an earlier release of z/VSE. In an online environment, DL/I VSE 1.12.1 can be used with CICS Transaction Server for z/VSE 2.2.

Features Introduced with DL/I VSE 1.12.1

DL/I VSE 1.12.1 introduces a partitioning function for HD (HDAM / HIDAM) databases. DL/I partitioning increases the maximum storage capacity of a DL/I database and overcomes the 4 GB limit for a single segment type.

DL/I Partitioning

A database can be divided into different sections that are called partitions, where each partition has its own set and the same number of data sets.

In Figure 1 on page 1, a nonpartitioned HD database that uses four data sets is compared to the same database that uses three partitions.

It is shown that by multiplying the number of data sets available in the nonpartitioned format by the number of partitions, one obtains the number of data sets available in the partitioned format.

The partition that is assigned to a database record is determined by the key value of its root segment. In the example, root key values can occur in the range '000000' to '999999' . The key ranges of the different partitions are defined by the user.

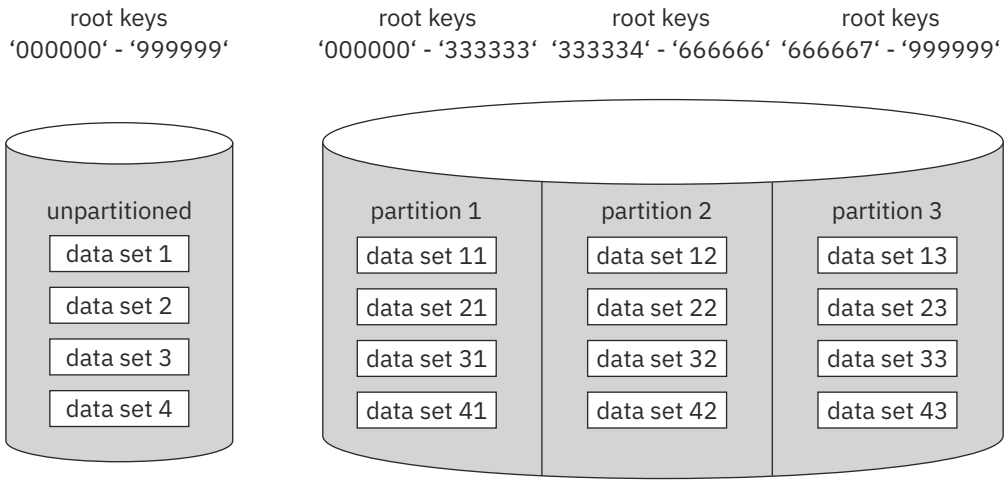

Figure 1. Partitioning a database

DL/I partitioning is transparent to user applications. Therefore, application programs do not have to be changed. It is not possible to process a partition individually.

Data Sets

Each partition owns the number of data sets as defined in the database description (DBD) with DATASET statements. In Figure 1 on page 1, four DATASET statements are assumed while three partitions are defined making 12 data sets available in total.

It is not possible to use different DATASET statements or a different number of DATASET statements for the different partitions.

Segment allocation

In a nonpartitioned database, DL/I assigns the segments of a database record to their data sets according to the order, in which DATASET and SEGM statements are following one another in the DBD. In the example of the nonpartitioned DBD of database STDCDBP shown in Table 1 on page 2.

- Root segment STSCCST is allocated on data set STDCDBC.
- Dependent segment STPCORD is allocated on data set ST2CDBC.

Table 1. Partitioning a database description (DBD)

database STDCDBP – unpartitioned DBD	database STDCDBP – partitioned DBD
DBD NAME=STDCDBP,...	DBD NAME=STDCDBP,...,NPARTS=(3,...
DATASET DD1=STDCDBC,...	DATASET DD1=STDCDB,...
SEGM NAME=STSCCST,...	SEGM NAME=STSCCST,...
...	...
DATASET DD1=ST2CDBC	DATASET DD1=ST2CDB
...	...
SEGM NAME=STPCORD,...	SEGM NAME=STPCORD,...

In a partitioned database, the same scheme applies. However, to use the appropriate version of the different data sets that are now available from one DATASET statement, DL/I assigns:

- A root segment to a data set with the suffix that corresponds to the partition that a specific routine has calculated for the value of its key as described in “Partition Selection” on page 3.
- A dependent segment to a data set with the suffix that corresponds to the partition calculated for its root.

In the example of the partitioned DBD of database STDCDBP shown in Table 1 on page 2, three partitions are defined.

The root segment STSCCST is allocated on data set:

- STDCDB1, if the value of its key falls within the key range of partition 1.
- STDCDB2, if the value of its key falls within the key range of partition 2.
- STDCDB3, if the value of its key falls within the key range of partition 3.

The dependent segment STPCORD is allocated on data set:

- ST2CDB1, if its root segment was assigned to partition 1.
- ST2CDB2, if its root segment was assigned to partition 2.
- ST2CDB3, if its root segment was assigned to partition 3.

Data set names

DBD generation suffixes the name that is entered on the DD1 parameter of a DATASET statement. For each DATASET statement, the different partition numbers create as many data set names and data set definitions as partitions have been defined. In Table 1 on page 2 the generated names are STDCDB1, STDCDB2, STDCDB3, ST2CDB1, ST2CDB2, and ST2CDB3.

The data set name that is entered through the DD1 parameter of a DATASET statement in the DBD of a partitioned database can have a maximum length of:

- Six characters, if up to 16 partitions are defined. DL/I uses the name from the DD1 parameter and suffixes it with “1”, “2”, “3”, ... “G”, corresponding to the number of the partition.
- Five characters, if more than 16 partitions are defined, up to the maximum of 128 partitions. DL/I uses the name from the DD1 parameter and suffixes it with:

- “11”, “12”, “13”, “14”, ... “1G” for the first 16 partitions
- “21”, “22”, “23”, “24”, ... “2G” for the second 16 partitions
- “81”, ”82”, “83”, “84”, ... “8G” for the last 16 of maximum 128 partitions.

At the end of the DBD generation listing, DL/I shows all data set names that have been created. These names must be used as // DLBL file names, which are then to be associated to the real file-ids of the VSAM ESDS clusters.

Data set attributes

Any DEVICE, BLOCK, SCAN, or FRSPC parameter that is entered on the first DATASET statement in a DBD stream is replicated to the data sets of any following DATASET statements. This includes the implicitly defined data sets generated for partitioning.

All VSAM define cluster attributes of the data sets must be identical.

Data set utilization

To obtain 12 data sets as shown in Figure 1 on page 1 one DATASET statement and 12 partitions could also be used, or 12 DATASET statements without partitioning if the number of segments that are defined allows this. However, when a database is defined as partitioned, a certain portion of the data blocks remains unused. The reason is:

1. DL/I rounds the length of segments including their prefix to align them on a certain boundary and the rounding increases with the number of partitions that are defined as shown in Table 2 on page 3.

Table 2. Database partitioning - segment lengths

up to # of partitions	segment length rounded to a multiple of	aligns segments to boundary
2	4	word
4	8	double-word (D-W)
8	16	DD-W
16	32	DDD-W
⋮	⋮	⋮

2. At the beginning of a block, DL/I rounds the offsets of root anchor points and the start of the segment space area to the same alignment that is in effect for segments.
3. At the end of a block, DL/I rounds the length of VSAM-related information to the same alignment that is in effect for segments.

Therefore, if database capacity needs to be extended, first increase the number of DATASET statements, then add partitioning.

Segment Lengths

DL/I rounds segment lengths to certain boundaries. However, this is transparent to application programs. The application programs see the original lengths as defined in the DBD or as passed by the user, in case of variable segment lengths.

Partition Selection

To obtain the partition, and thereby the data sets where the segments of a database record are to be stored, DL/I calls a partition selection routine, which must be provided by the user. The partition selection routine calculates the partition number of a given root segment based on the key value passed for that root.

The interface between DL/I and the partition selection routine is the partition selection control area (PSCA), which holds the following information:

Input

- Name of partition selection module
- Database name
- Root key length-1
- Entry point of partition selection module
- Address of first byte of root key
- Number of partitions defined

Output

- Calculated number of partition

The new DL/I member DLZPSL10 is a sample partition selection routine, which selects between two key ranges to determine the partition number that is returned on a passed root key value. DLZPSL10 can also be used as a sample to write a custom partition selection routine.

Definition

DL/I partitioning is defined with the new NPARTS parameter of the DBD macro.

NPARTS=(*nnn*,*name*)
nnn specifies the number of partitions. The value of *nnn* can range from 2 - 128.

name specifies the name of the partition selection routine. It determines in which partition a database record is stored and must be provided by the user.

For example: `DBD NAME=STDCDBP,ACCESS=HIDAM,NPARTS=(3,DLZPSL10)`

The next steps that are required are:

1. DBDgen (assembly and link-edit) of the changed DBD.

   A new PSBgen is not required.
2. ACBgen with parameter `DMB=YES`

   on all PSBs containing a PCB that represents or references a partitioned database.

This creates new DMB and PSB phases.

Note: Whenever the number of partitions or the partition selection routine are changed for an existing partitioned database, it is necessary to run a DL/I re-organization.

Loading the Database

A database can be loaded in partitioned format in the following ways:

- The database can be initially loaded with a regular DL/I application program using the newly built database definitions and partition selection routine.
- An existing database can be converted from unpartitioned to partitioned format. This is described in detail in “Migration” on page 4.

Migration

To convert a database to partitioned format, a DL/I reorganization is necessary. The required steps are:

1. DL/I Unload (DLZURGU0) of the existing database using old DMB / PSB phase.

2. Adapt database definition using new NPARTS parameters on DBD statement and, if needed, adjusted lengths of DD1 names on DATASET statements.
3. Run DBDgen of the changed DBD.
4. Run ACBgen on all affected PSBs with parameter DMB=YES to create new DMB / PSB phases.
5. Provide partition selection routine.
6. Delete / Define VSAM clusters for all required data sets.
7. DL/I Reload (DLZURGL0) using new DMB / PSB phase.
8. If secondary indexes or logical relationships exist, DL/I Prefix Resolution (DLZURG10) and DL/I Prefix Update (DLZURGP0) must be run.

Operation

DL/I partitioning is transparent to user applications. However, when sequentially reading a partitioned HDAM database, it is possible that records are returned in a different order than when sequentially reading the same database in unpartitioned format. The reason is that records are now read as additional and first order, partition by partition.

This is not the case with partitioned HIDAM databases. There, records are returned as before in root key sequence during sequential reads. This is independent of partitions.

Utilities

All DL/I utilities can be used in the same way as before when a database is defined as partitioned. However, control statements might have to be changed or added to address the new data sets.

Note: Partial database load and Partial database reorganization are not supported for partitioned databases.

Enhancements introduced through APAR fixes in DL/I VSE 1.12.1

The following enhancements are available for DL/I VSE 1.12.1 as APARs. The corresponding APAR numbers are shown in parentheses.

DL/I Utility Improvements for Database Recovery (PH29950)

New functions have been provided for the DL/I Log Print utility and the DL/I Data Base Backout utility to facilitate the recovery of a DL/I database.

Stop Log Print on Log Sequence Number

A new parameter has been introduced for the DL/I log print utility (DLZLOGP0) that allows to stop printing of log records, when a DL/I log record with a defined sequence number is reached.

This function might be needed during DL/I database recovery and only log records that had been written up to a defined point should be applied.

To provide this function, the following new parameters can be entered on the LO input control statement:

Column	Description
54-61	1-8-digit DL/I log sequence number as shown with the SEQNO-keyword in the DL/I log records. It must be left-aligned or supplied with leading zeros and contain only hexadecimal digits (0-9, A-F). If specified, printing of log records stops, when a DL/I log record containing this sequence number has been found. If omitted, all log records will be printed unless excluded by another parameter on the LS or LO statement.

Column	Description
63-64	1-2-digit version of the DL/I log sequence number. It must be left-aligned or supplied with leading zero and be in the range 1-99. The default value is 1. If specified, printing of log records stops, when the n-th occurrence of the specified sequence number has been found, where 'n-th' corresponds to the value entered for this parameter.

Note: If the new keyword CPYR described in "Log Print with Tape-Copy-by-Record (PM56161)" on page 7 has been specified, copying of log records is stopped as well at the defined DL/I log sequence number / version.

When a DL/I log sequence number / version has been specified and is found before the end of the log input stream, the new message

DLZ462I END SEQUENCE NUMBER EXCEEDED ON FILENAME – *file name*

is written.

In addition, if there are scheduling records without a matching termination record at this point, existing message

DLZ424I NO LOG TERMINATION RECORD FOUND FOR SCHEDULING RECORD

is written, together with existing, and extended message

```
DLZ425I PSBNAME= psbname,SEQUENCE NUMBER=nnnnnnnn
        DMBNAMEs=dmbname1,dmbame2, …
```

Psbname is the name of the PSB and *nnnnnnnn* the DL/I log sequence number of the scheduling record, for which the log print utility did not find a matching termination record.

The new 'DMBNAMEs =' line(s) show the names of all databases (DMBs) that were changed – that is, for which DL/I data base-type log records had been written – since the PSB has been scheduled at the record with the indicated log sequence number, or since the last DL/I checkpoint had been written for this PSB (batch execution only).

Backout per Database in Batch

The DL/I data base backout utility (DLZBACK0) is invoked with the name of the PSB, for which backout is to be executed. This removes inflight changes from all databases that are referenced by this PSB and for which records are found on the log data set.

During DL/I database recovery it may be needed not to back out all databases for a PSB, but only one or a subset of them.

To provide this function, the new LS input control statement has been introduced that can be entered in the following way:

Column	Description
1-2	Must be LS.
4-10	Database name.

Each LS statement defines one database of the PSB, for which backout is to be done. Up to 16 different database names can be entered. Without the LS statement, backout is executed for all databases of a PSB as before.

DL/I VSE 1.12.0 Enhancements

This section introduces the enhancements that are available with DL/I VSE 1.12.1. They have also been made available with APAR fixes for DL/I VSE 1.12.0. The corresponding APAR numbers are shown in parentheses.

Wrong reply to message DLZ142A may corrupt database (PM57974)

If an I/O error, which is shown through message DLZ004I or DLZ005I, occurs during CICS/DLI operation, the following CICS emergency restart writes messages DLZ141I and DLZ142A to the console prompting the user to reply with CONTINUE, IGNORE, or CANCEL.

If the reply is CONTINUE, but the affected database had not been repaired, as described in Chapter 9, “Recovery after a DLZ004I or DLZ005I Error,” on page 69, the database might become corrupted.

To make the user visually aware of the potential risk of an inappropriate reply, message DLZ142A has been extended to include a warning:

```
DLZ142A ENTER CONTINUE (ONLY IF DATABASE RECOVERED), IGNORE OR CANCEL
```

Log Print with Tape-Copy-by-Record (PM56161)

When recovering a DL/I database, it may be needed to apply only those log records that had been written during a certain time interval. This option is available, for example, if a new log file that contains only records that had been written after or before a defined point is created out of existing log files. To provide this function, the new keyword **CPYR** has been introduced for the DL/I log print utility. It can be specified on columns 41-44 of the LO control statement. **CPYR** recognizes all filters that can be entered via parameters of the LO and LS control statements. This includes a start / end date on columns 13-21 / 23-31, or, as introduced in “Stop Log Print on Log Sequence Number” on page 5, a log sequence number / version on columns 54-61 / 63-64 of the LO control statement. If **CPYR** is used, the same records from the input log files that are going to be printed are also copied to a new output log tape, with this exception:

- CICS (that is, non-DL/I) log records are always copied to the output tape, independent of any filters and also if parameter **CICSD** was not set on columns 46-50 of the LO control statement.

The original COPY function (keyword 'COPY' on columns 41-44 of the LO control statement) works as before and unconditionally copies over all log records.

Support up to 16 data sets for DL/I HD databases (PM71542)

Starting with DL/I VSE 1.11 a DL/I HD database can be allocated on up to 5 VSAM ESDS data sets. APAR PK48347 extended this to 10 data sets and PM71542 to 16 data sets.

Migrating to DL/I VSE 1.12.1

DL/I VSE 1.12.1 is upward compatible from DL/I VSE 1.12.0. Programming interfaces are not changed. DL/I batch and online applications should run as before.

Online users must reassemble and re-create a new DL/I online nucleus by using the DLZACT generation procedure.

Refer to “DL/I Partitioning” on page 1 for migration aspects that are related to the new support of partitioned databases.

If you are migrating from an earlier DL/I release than DL/I VSE 1.12.0, refer to Chapter 4, “Migration and Compatibility Considerations,” on page 15.

New and Changed Messages

Refer to Chapter 10, “Messages and Codes,” on page 93 for details.

Chapter 2. Operating Environment

This section describes the machine and programming requirements as well as the programming environment for the operation of DL/I VSE 1.12.1.

Machine Requirements

Virtual Storage

The minimum main storage requirements for DL/I VSE 1.12.1 are the same as those needed for conventional operation of your VSE system. Because of virtual storage support, the minimum configuration is dependent on application characteristics and performance requirements. Processor performance may be traded off against main storage size.

For additional details regarding storage needs, refer to DL/I Data Base Administration, in:

"Part: Installation Planning"
"Appendix: DL/I Virtual Storage Estimate"
"Appendix: DL/I Real Storage Estimate"

Disk Storage

The approximate library space (in library blocks) needed for a DL/I VSE 1.12.1 installation is given below. Either check that an existing sublibrary has enough free library blocks, or allocate a new library of at least the indicated size. It is advisable to allocate 5% to 10% additional space to the values given below for adding new code and applying service.

```
Production Sublibrary:  6.800 library blocks
Generation Sublibrary: 12.900 library blocks
```

To find out the actual amount of library space required for the installation of DL/I VSE 1.12.1, scan the distribution tape using the VSE-provided installation dialog.

Tape Drive

You might need a tape drive depending on how you want to install DL/I VSE 1.12.1 or apply service to it.

Using a Tape as Logging Device

When you use a tape device as DL/I log, where either buffered or unbuffered write mode can be selected (for example, the IBM 3480, IBM 3490, IBM 3590, IBM 3592, or a later model) you have to use *Tape Write Immediate* (non-buffered) mode. This is for reasons of data integrity because it ensures that data in the buffer is not lost if a power interruption or system failure occurs.

Also, the CICS Transaction Server supports tape devices with this characteristic. However, these units are not recommended for journaling.

Terminal Support

Any terminal device supported by CICS, or equivalent product, can be used for online DL/I. Use of the DL/I IMF and IUG functions requires a 3270-type terminal.

Running DL/I in a VM Environment

If you plan to run VSE under VM, you have to consider VM storage requirements.

For using DL/I with CICS in a VM virtual machine (under control of VSE), the following considerations apply:

- CICS when operating in a virtual machine has the same hardware and software requirements as CICS operating in a real machine.
- The minimum hardware requirements of CICS should be considered as additional to the minimum requirements for VM itself and any other virtual machines within the VM environment.
- CPU utilization and possibly terminal response times will be greater when CICS is running under VM than when it is running in a real machine environment. The effect on performance will be most noticeable when VM is introduced into an installation where CPU and main storage resources are already substantially committed to existing CICS and other work.

Programming Requirements

The following IBM programs are required for the operation of DL/I VSE 1.12.1:

- z/VSE 6.2 (5686-VS6)
- DFSORT/VSE 3.4 (5746-SM3) (or equivalent)

Programming Environment

DL/I VSE 1.12.1 is designed to work with the following IBM programs:

- **z/VSE 6.2 Base Programs**
  - CICS Transaction Server for z/VSE 2.2
  - VSE/VSAM Space Management for SAM
  - VSE/ICCF

    DL/I batch and MPS batch jobs can run in VSE/ICCF interactive partitions.
- **z/VSE Extended Base Program**
  - DB2 Server for VSE & VM 6.1 or later
- **VM/SP 3.1 or later**
  - VM (including VM/XA, VM/ESA and z/VM) supports the DL/I IMF and IUG functions through ISPF.

Compiler and Assembler Support

DL/I VSE 1.12.1 supports the following compilers and assemblers:

- High Level Assembler for VSE 1.6
- LE/VSE 1.4
- COBOL for VSE/ESA 1.1
- PL/I for VSE/ESA 1.1
- DOS/VS RPG II 1.3

Chapter 3. Installation

Note: DL/I VSE 1.12.1 requires z/VSE 6.2. It cannot be used with z/VSE 6.1 or an earlier release of z/VSE. To install DL/I VSE 1.12.1 into the correct sublibraries, make sure that PTF UI56255 has been applied.

DL/I VSE 1.12.0 can also be used with z/VSE 6.2.

VSE provides dialogs to install optional programs. To install DL/I, use the dialog *Install Program(s) from Tape*. You reach this dialog via the *Install Programs-V2 Format* dialog. The dialogs are described in detail in z/VSE Installation.

Before installing DL/I VSE 1.12.1, first remove any older release of DL/I (DL/I DOS/VS 1.10, DL/I VSE 1.11, or DL/I VSE 1.12.0) that still resides on your system. You can do this by using the Delete Book DLZDLETE that is provided in the DL/I production sublibrary. Follow the instructions included in this member to build the appropriate delete job.

Note: For using the correct version of DLZDLETE, PTF UK59700 (DL/I DOS VS 1.10) or PTF UK59701 (DL/I VSE 1.11) must be installed. No PTF is needed for DL/I VSE 1.12.0.

If you install DL/I VSE 1.12.1 before removing a previous release of DL/I, the following message is displayed:

```
M231D INSTALLATION WILL OVERWRITE PRODUCT … ENTER "DELETE" OR "KEEP”
```

In this case, it is recommended to do the following:

- Cancel the installation job.
- Run the appropriate delete jobs to remove previous DL/I systems.
- Resubmit the installation job.

DL/I Shipment Details

DL/I VSE 1.12.1 is shipped as VSE optional program.

Note: Starting with z/VSE 6.2 physical tape delivery has been dropped. The former optional product tapes or cartridges are provided as distribution tape images in AWS format. They can be obtained on DVD or downloaded from the Internet.

The distribution tape image contains a complete set of (or replacement for) the DL/I libraries, including all IMF and IUG functions, panels, skeletons, and message files.

Note: DL/I IMF and IUG is shipped in its VM version only. Refer also to “DL/I IMF and IUG Functions” on page 21.

DL/I is partially delivered in object code and is shipped with a production and a generation sublibrary.

Production Sublibrary

The DL/I production sublibrary contains:

- All DL/I phases (type PHASE)
- All DL/I modules (type OBJ)
- All primary source code of DL/I, consisting of:
  - Source books of type A
  - Sample programs of type A, C, P, and R (see “Source Books in the Production Sublibrary” on page 12).

Source Books in the Production Sublibrary

The DL/I production sublibrary contains the following source books needed for certain tasks:

Structure for User Interface Blocks

DLIUIB.C
User Interface Block for COBOL

DLIUIB.P
User Interface Block for PL/I

DLIUIB.R
User Interface Block for RPG

DLIAIB.C
Application Interface Block for COBOL (AIBTDLI i/f)

DLIAIB.P
Application Interface Block for PL/I (AIBTDLI i/f)

DL/I Database Definition and Load Samples

DLZMAP.A
Mapping module for DLZSAM60

DLZSAMAC.A
Batch ACCESS sample application job stream

DLZSAMJS.A
Online sample application job stream

DLZSAM40.A
Load program

DLZSAM50.A
Batch print program

DLZSAM60.A
Online sample application program (not supported with CICS Transaction Server)

DL/I High Level Assembler Sample Programs

DLZHMAP.A
Mapping module for DLZHLA60

DLZHLA40.A
Load program

DLZHLA50.A
Batch print program

DLZHLA60.A
Online sample application program

DLZHLA80.A
Sample program using the AIBTDLI interface

DLZPSL10.A
Sample partition selection routine

DL/I COBOL Sample Programs

DLZCBMAP.A
Mapping module for DLZCB230, DLZCB260

DLZCB210.A
HLPI load program

DLZCB220.A
HLPI batch print program

DLZCB230.A
Online HLPI sample application program

DLZCB240.A
Load program

DLZCB250.A
Batch print program

DLZCB260.A
Online sample application program

DLZAIC50.A
Sample program using the AIBTDLI interface

DL/I PL/I Sample Programs

DLZPLMAP.A
Mapping module for DLZPL230, DLZPL260

DLZPL210.A
HLPI load program

DLZPL220.A
HLPI batch print program

DLZPL230.A
Online HLPI sample application program

DLZPL240.A
Load program

DLZPL250.A
Batch print program

DLZPL260.A
Online sample application program

DLZAIP50.A
Sample program using the AIBTDLI interface

DL/I RPG Sample Programs

DLZRGMAP.A
Mapping module for DLZRPG60

DLZRPG40.A
Load program

DLZRPG50.A
Batch print program

DLZRPG60.A
Online sample application program

DL/I Documentation Aid, ISQL EXTRACT DEFINEs

DLZDATAB.A
Acquires DB space and creates the SQL/DS DL/I DA tables

DLZDANDX.A
Creates the SQL/DS DL/I DA table indexes

DLZDARTN.A
DATALOADs the ISQL DL/I DA routines

DLZDLBD.A
Creates the DBD ACCESS module

DLZDLBP.A
Creates the PSB ACCESS module

DLZEXDF.A
Job stream for installing ISQL EXTRACT DEFINEs

DLZEXWCB.A
EXTRACT DEFINEs work control block

DLZSQLID.A
USERID control block

Special DL/I Books

DLZACTDS.A
DSECT for application control table (ACT)

DLZDBGLB.A
COPY book for DBD global values

DLZDLETE.A
DL/I delete book

DLZLNKBK.A
DL/I link book

DLZMERGE.A
DL/I merge book

Generation Sublibrary

The DL/I generation sublibrary contains:

- Macros for DBD, PSB, ACB and online generation.
- The VM Version of IMF and IUG (DLZCMSTL.Z and DLZCMSML.Z).
- The optional source code of DL/I (type A).

Chapter 4. Migration and Compatibility Considerations

This section discusses considerations for users who want to migrate from a previous DL/I release.

For the latest migration information, refer to “Migrating to DL/I VSE 1.12.1 ” on page 7.

Depending on your current DL/I release, select the appropriate entry point:

- **1.7.0**

  Refer to “Migration from DL/I 1.7.0 to 1.7.1” on page 15.
- **1.7.1**

  Refer to “Migration from DL/I 1.7.1 to 1.8.0” on page 15.
- **1.8.0**

  Refer to “Migration from DL/I 1.8.0 to 1.9.0” on page 16.
- **1.9.0**

  Refer to “Migration from DL/I 1.9.0 to 1.10.0” on page 16.
- **1.10.0**

  Refer to “Migration from DL/I 1.10.0 to 1.11.0” on page 16.
- **1.11.0**

  Refer to “Migration from DL/I 1.11.0 to 1.12.0” on page 16.
- **1.12.0**

  Refer to “Migration from DL/I 1.12.0 to 1.12.1” on page 17.

Furthermore, this section discusses the compatibility of DL/I with IMS/VS.

Migration from DL/I 1.7.0 to 1.7.1

DL/I 1.7.1 is upwardly compatible from DL/I 1.7.0.

Changes are required to DL/I user application programs only for the following cases:

- The DL/I call function GSCD is supported only for applications operating in the same non-shared address space as DL/I. Applications operating in address space other than the one in which DL/I and CICS are operating must be converted to use the new GSTA call to retrieve the buffer statistics.
- ACBGEN must be rerun for any PSBs for which you wish to issue the new GSTA calls.
- "PATH" sensitivity must be specified for all segments for which you do path inserts from MPS jobs running in separate address spaces. Currently, "PATH" sensitivity is required only for path retrieves but not for path inserts. (This is also a requirement for users of remote PSB support.)

Except for the above mentioned, no other ACBGEN, DBDGEN, or PSBGEN run is required for existing programs.

Continue the migration with “Migration from DL/I 1.7.1 to 1.8.0” on page 15.

Migration from DL/I 1.7.1 to 1.8.0

DL/I 1.8.0 is upwardly compatible from DL/I 1.7.1.

Changes are required to DL/I user application programs only for the following cases:

- All online users must reassemble the online nucleus with the DL/I DLZACT macro.

- All programs which contain DLZTRACE macro calls with OUTPUT=SYSLST must be reassembled and re-linked.
- All user-written modules or programs (for example, randomizing routines) which reference the Partition Specification Table (PST) must be reassembled and re-linked.

Continue the migration with “Migration from DL/I 1.8.0 to 1.9.0” on page 16.

Migration from DL/I 1.8.0 to 1.9.0

DL/I 1.9 is upwardly compatible from DL/I 1.8.0.

Changes are required to DL/I user application programs only for the following cases:

- All online users must reassemble the online nucleus with the DL/I DLZACT macro.
- All online users must reassemble and re-link their online trace programs using the DLZTRACE macro.

Continue the migration with “Migration from DL/I 1.9.0 to 1.10.0” on page 16.

Migration from DL/I 1.9.0 to 1.10.0

DL/I 1.10 is upwardly compatible from DL/I 1.9.0.

If you are an online user migrating from DL/I 1.9 to DL/I 1.10, you have to reassemble the online nucleus with the DL/I DLZACT macro.

If you want to exploit the support provided with DL/I 1.10, you have to make changes to your existing procedures as described below.

VSE/VSAM HS-Buffers above the 16MB Line of Storage

If you want to allocate VSE/VSAM input/output buffers above the 16MB line of storage for:

- KSDS index files, or
- HISAM KSDS and ESDS data files, or
- SHISAM KSDS data files,

proceed as explained under “VSE/VSAM HS-Buffers Above the 16 MB Line of Storage” on page 42.

Virtual Disk Exploitation

If you want the work files of DL/I utilities to reside on virtual disk instead of on a real device, the job control statements (ASSGN, DLBL and EXTENT) must address the virtual disk instead of the real device.

For more information, refer to “Virtual Disk Exploitation” on page 44.

Continue the migration with “Migration from DL/I 1.10.0 to 1.11.0” on page 16.

Migration from DL/I 1.10.0 to 1.11.0

Refer to “Migrating to DL/I VSE 1.11 and the CICS Transaction Server for VSE/ESA 1.1” on page 56.

Continue the migration with “Migration from DL/I 1.11.0 to 1.12.0” on page 16.

Migration from DL/I 1.11.0 to 1.12.0

Refer to “Migrating to DL/I VSE 1.12.0 ” on page 65 .

Continue the migration with “Migration from DL/I 1.12.0 to 1.12.1” on page 17.

Migration from DL/I 1.12.0 to 1.12.1

Refer to “Migrating to DL/I VSE 1.12.1 ” on page 7.

Compatibility with IMS/VS

DL/I is compatible with IMS/VS for the batch user and to IMS/VS with a CICS subsystem for the online user, with the following exceptions:

- The following facilities in DL/I are not supported by IMS/VS:
  - HD or HISAM UNLOAD in IMS/VS and RELOAD in DL/I VSE
  - RPG-II support
  - FBA devices
  - CKD device independence
  - CALLDLI MF=E Assembler Language DL/I programs
  - IMF and IUG
  - Disk logging
  - Application control table (ACT)
  - ACCESS macro
  - Extended remote PSB
  - Selective UNLOAD
  - CICS Monitoring Facility (CMF) DL/I PA replacement
  - Run and buffer statistics
  - Checkpoint option for HD UNLOAD
  - Rewind option in utilities
  - Documentation Aid
  - MPS Restart
  - Variable-length index source segments
  - PL/I options
  - GSTA call
- The following DL/I facilities are not fully compatible or are supported by IMS/VS in a different manner and require some user modifications:
  - High level programmer interface (IMS/VS)
  - Log files
  - Image copy files
  - Index format
  - Field level sensitivity
  - HD UNLOAD in DL/I VSE and HD RELOAD in IMS/VS
  - HISAM UNLOAD in DL/I VSE and HISAM RELOAD in IMS/VS
  - UNLOAD/RELOAD disk files
  - CKPT CALL parameters
  - MODEL=3330-II
  - Default PSB on scheduling call
  - DL/I status code NI
  - Control Interval sizes
  - Partial Data Base Load

For further information on IMS/VS compatibility refer to DL/I Data Base Administration, in "Appendix: Incompatibilities between DL/I and IMS".

Chapter 5. Customizing DL/I

SVA Loading for DL/I

The following DL/I phases are eligible for residence in the SVA:

$SVADLI
SVA Load List

DLZCPY10
Field Level Sensitivity Copy

DLZDBH00
Buffer Handler

DLZDLA00
Call Analyzer

DLZDLD00
Delete/Replace

DLZDLR00
Retrieve

DLZDDLE0
Load/Insert

DLZDHDS0
Space Management

DLZDXMT0
Index Maintenance

DLZSTRB0
Batch Field Storage Manager

Together, these DL/I phases require a physical size of at least 96K in the SVA. You have to reserve the space during system start-up by using the IPL command:

```
SVA ....,PSIZE=nnnK,...
```

You can load all or a selection of these phases into the SVA. Proceed as shown under:

- “Loading all SVA Phases” on page 19
- “Loading User-Specified SVA Phases” on page 20
- “Loading SVA Phases via IPL” on page 20

Loading all SVA Phases

To load all SVA-eligible phases, use the SVA load list $SVADLI and run a job similar to that shown in Figure 2 on page 19.

```
// JOB SVALOAD1
// LIBDEF PHASE,SEARCH=DLIPRD.sublib
SET SDL
LIST=$SVADLI
/*
/&
```

Figure 2. Sample Job SVALOAD1

Loading User-Specified SVA Phases

To load selected SVA-eligible phases, run a job similar to that shown in Figure 3 on page 20.

```
// JOB SVALOAD2
// LIBDEF PHASE,SEARCH=DLIPRD.sublib
SET SDL
DLZCPY10,SVA
DLZDBH00,SVA
DLZDLA00,SVA
DLZDLD00,SVA
DLZDLR00,SVA
DLZDDLE0,SVA
DLZDHDS0,SVA
DLZDXMT0,SVA
DLZSTRB0,SVA
/*
/&
```

Figure 3. Sample Job SVALOAD2

Loading SVA Phases via IPL

To load SVA-eligible DL/I phases automatically each time IPL is performed, you may incorporate the appropriate job control statements into your BG ASI procedure $0JCL.

Depending on whether you want to include all or user-selected phases, use the job control statements shown in:

- Figure 2 on page 19
- Figure 3 on page 20

In the load procedure, omit the // JOB and /& statements.

SVA Phases Needed for the AIBTDLI Interface

When you are running programs using the AIBTDLI interface, you may additionally want to load these phases into the SVA:

DLZMPX00
AIBTDLI Connector Interface

DLZBSEOT
MPS Task Exit Routine

CICS Adaptation

If you plan to:

- run DL/I online, or
- employ MPS,

ensure that the interface between DL/I and CICS is correct by performing the following steps:

1. Define DL/I databases and application programs:
   a. Define DL/I databases in the CICS file control table (FCT). Include an entry for each database description (DBD) corresponding to a physical database. The name in the DATASET parameter in the FCT and the NAME parameter in the DBD must be identical.
   b. Define the applications that access the DL/I databases in your CSD file. Specify TASKDATAKEY(USER) and EXECKEY(USER).
2. Create DL/I and CICS system tables for DL/I support as follows:
   a. Define the DL/I application control table (ACT). This table is required to associate online application programs with one or more DL/I databases.

With DL/I VSE 1.11, support of the Storage Layout Control (SLC) table has been dropped.

b. In a CICS XRF environment, if DL/I program isolation is active, or CICS emergency restart or dynamic transaction backout is to be used with DL/I tasks, assign the DL/I log to the CICS system log. In this case, specify DBP=2$ in the CICS System Initialization Table (SIT). The DL/I log is assigned by use of the VSE UPSI byte information. For information on how to use this UPSI byte with CICS refer to:

   - "UPSI Byte Settings (Online)" in the DL/I DOS/VS User's Guide.

c. If you want to capture DL/I run and buffer statistics, you must include the DL/I module DLZSTTL in your resource definitions. The run and buffer statistics function captures online DL/I system statistics and writes them to the CICS extra-partition destination CSSL. This data is automatically printed during CICS shutdown, or printing can be invoked by the CSDE transaction.

   You can add the DL/I run and buffer statistics function to your CICS system by defining:

   - DLZSTTL as program, and
   - CSDE as transaction in the CSD file.

d. The CICS Monitoring Facilities (CMF) let you, as a CICS DL/I user, collect performance data during online processing. For detailed information, refer to "CICS/VS Monitoring Facilities (CMF) Hooks" in DL/I Data Base Administration.

A CICS-DL/I installation tester application program (DFHTDLI) is supplied with the pre-generated CICS system. For the description of the program, refer to:

- CICS Transaction Server System Definition Guide

For a full description of all DL/I related CICS tables, refer to DL/I Resource Definition and Utilities.

For a description of how to use DL/I and CICS XRF, refer to “Features Introduced with DL/I DOS/VS 1.9” on page 39.

DL/I IMF and IUG Functions

Note: DL/I IMF and IUG needs ISPF as a prerequisite. While ISPF is not supported anymore since VSE/ESA, you can use DL/I IMF and IUG functions when ISPF is running under VM/CMS. The VSE version of DL/I IMF and IUG is no longer shipped with DL/I VSE 1.12.

DL/I provides two interactive functions: the Interactive Macro Facility (IMF), and the Interactive Utility Generation (IUG) facility. These functions offer easy-to-use interactive procedures that let you perform resource definition and utility functions at a terminal. For a full description of these DL/I functions, refer to *DL/I Interactive Resource Definition and Utilities*.

The Interactive System Productivity Facility (ISPF) handles the menus and dialogs for DL/I IMF and IUG.

Note: If you intend to use IMF tables which were originally created under IPF (Interactive Productivity Facility) with a DL/I release prior to DL/I 1.7, these tables must be converted to ISPF format. To perform the conversion, refer to *DL/I Interactive Resource Definition and Utilities*, in "Appendix: Converting DL/I Tables from IPF to ISPF Format".

Using IMF and IUG Functions in a VM Environment

If you plan to use the IMF and IUG functions under VM/CMS, you can install DL/I IMF and IUG functions on VM for execution with ISPF. To use these functions, you need 4.3 MB of disk space.

Installing DL/I Dialogs on CMS

To install the IMF and IUG members, perform the following steps.

1. Start up your VSE system and run the following VSE job:

```
// JOB CMS
// EXEC LIBR
  ACCESS SUBLIB=DLIGEN.sublib
```

```
      PUNCH DLZCMSTL.Z
      PUNCH DLZCMSML.Z
    /*
    /&
```

2. Route the punch output of the above job to a CMS userid, for example by issuing the CP spool command:

```
SPool PUNch userid
```

3. When the punch output of the VSE job is in your reader, receive it, for example by issuing the CMS command:

```
   READCARD * * A
```

   Ignore or erase the resulting CMS file:

```
   READCARD CMSUT1 A
```

4. The output of this reading are the following VM files (they represent the VM version of IMF and IUG):

```
DLZ0TEXT TXTLIB A
DLZPANL1 MACLIB A
DLZPANL2 MACLIB A
DLZPANL3 MACLIB A
DLZSKELS MACLIB A
DLZMSGS MACLIB A
```

5. Create and run a CMS EXEC containing all FILEDEFs for ISPF.
6. Enter `ISPSTART PANEL(DLZ0)` in CMS to access the IMF and IUG main menu.

For additional information, refer to *DL/I Interactive Resource Definition and Utilities*.

Deleting the VM Version of IMF and IUG

If you do not plan to use the IMF and IUG functions under VM/CMS, you may want to delete the IMF and IUG members from your DL/I libraries to save space. To do this, run the job stream shown in Figure 4 on page 22.

```
// JOB DELETVM
// EXEC LIBR,PARM='MSHP'
ACCESS SUBLIB=DLIGEN.sublib
DELETE  DLZCMSTL.Z  DLZCMSML.Z
/*
/&
```

Figure 4. Sample Job DELETVM

DL/I Documentation Aid and the ISQL EXTRACT DEFINEs Utility

Note: The following information on the DL/I Documentation Aid facility refers to SQL/DS. It also applies to the VSE Extended Base Program DB2 Server for VSE & VM to the extent that DB2 is upward compatible to SQL/DS. For the ISQL EXTRACT facility, you need SQL/DS prior to Version 3.

The DL/I Documentation Aid (DA) facility is an ease-of-use facility to document DL/I Data Base Description (DBD) and Program Specification Block (PSB) definitions that can be accessed directly by ISQL. This allows monitoring the DL/I database definitions. This facility is available to DL/I users who have SQL/DS and ISQL installed on their VSE system. When the Documentation Aid is invoked by the DL/I Application Control Blocks Creation and Maintenance utility, all the DBD and PSB definitions are automatically recorded into a special group of SQL/DS tables.

The ISQL EXTRACT DEFINEs utility provides the automatic creation of an ISQL routine of EXTRACT DEFINE commands, eliminating the repetitive task of identifying the DL/I databases to the ISQL EXTRACT facility. This utility obtains its definitions from the SQL/DS tables created by the DL/I Documentation Aid.

Before using the DL/I Documentation Aid facility and DL/I EXTRACT DEFINEs, you should be familiar with the following publications:

- DL/I Resource Definition and Utilities, in
  - "Chapter: Defining a Program Specification Block"
  - "Chapter: Doing the ACBGEN Procedure"
- *SQL/Data System Administration for VSE*.

Note: Support of DL/I Extract has been removed by SQL/DS Version 3.

DL/I Documentation Aid

Listed are the steps to install and activate the DL/I Documentation Aid (DA):

1. Create and run a job similar to job PUNCHDA, (Figure 5 on page 23). This job punches three DL/I Documentation Aid job streams (provided in the production sublibrary) using the VSE librarian. Provide the appropriate job control information where indicated. The purpose of each job stream is:

   DLZDATAB
   Acquires DB space and creates the SQL/DS DL/I DA tables.

   DLZDANDX
   Creates the SQL/DS DL/I DA table indexes.

   DLZDARTN
   DATALOADs the ISQL DL/I DA routines in the routine table.

   ```
   // JOB PUNCHDA
   // EXEC LIBR
   ACCESS SUBL=DLIPRD.sublib
   PUNCH DLZDATAB.A
   PUNCH DLZDANDX.A
   PUNCH DLZDARTN.A
   /*
   /&
   ```

   Figure 5. Sample Job PUNCHDA

2. Remove all the CATALOG and ,'+ statements from each of the three Documentation Aid job streams.
3. Ensure that the final two statements in each job stream are ,'* and ,'&.
4. Preprocess and reassemble the modules DLZDLBD and DLZDLBP with the SQL/DS release installed on your system.

   To do this, create and run a job similar to that shown in Figure 6 on page 25. This job stream creates the type A, type OBJ, and type PHASE entries for the modules.

   Figure 6 on page 25 shows processing for module DLZDLBD. For the module DLZDLBP, change the names according to Table 3 on page 24.

   Note: If during assembly of DLZDLBDP and DLZDLBPP, message ASMA044E "Undefined symbol - SQLDSIZ" occurs, remove the END statement in the last line of DLZDLBDP and DLZDLBPP, or put it at the source end.

   After this, or in general, whenever DLZDLBDP or DLZDLBPP have been reassembled, the following other modules of the DL/I Block Builder Utility DLZUACB0 have also to be reassembled and their objects cataloged in the DL/I production sublibrary:

   - DLZDLBL0
   - DLZDLBL1
   - DLZDLBL2
   - DLZDLBL3

   Then the DL/I Block Builder utility DLZUACB0 has to be re-linkedited from the new objects.
5. All job control statements to be executed are set to invoke SQL/DS in single-partition mode.

If you are running in:

- Single-partition mode, update the EXEC PROC= statement to include your installation's procedure name for the SQL/DS database identification statements.
- Multi-partition mode, delete the EXEC PROC= statement and change the EXEC statement for a multi-partition environment.

6. Update the password in all EXEC and SQL/DS CONNECT statements.
7. Remove the UPSI statement from the CRTDLBDP job if you want both punched and printed output.
8. Change the ACQUIRE statements in DLZDATAB only if it is necessary to increase your database space (DBSPACE).
9. Run the output (DLZDATAB, DLZDANDX, DLZDARTN) acquired in Step 1, plus the job shown in Figure 6 on page 25.
10. After running all job streams, you can use the DL/I Documentation Aid.

ISQL EXTRACT DEFINEs

To use the ISQL EXTRACT facility, you need an SQL/DS prior to version 3.

You must first preprocess and reassemble the module DLZEXDF with the SQL/DS release installed on your system. This must be done according to the naming convention shown in Table 3 on page 24 and using the sample job shown in Figure 6 on page 25.

Note: If during assembly of DLZEXDFP, message ASMA044E "Undefined symbol - SQLDSIZ" occurs, remove the END statement in the last line of DLZEXDFP, or put it at the source end.

Table 3. Naming Convention for DL/I DA and ISQL EXTRACT Modules

Module to be Preprocessed	Source to Catalog	Object Code to Catalog	Phase Name
DLZDLBD.A DLZDLBP.A DLZEXDF.A	DLZDLBDP.A DLZDLBPP.A DLZEXDFP.A	DLZDLBDP.OBJ DLZDLBPP.OBJ DLZEXDFP.OBJ	DLZDLBDP.PHASE DLZDLBPP.PHASE DLZEXDFP.PHASE

Note that in Figure 6 on page 25:

- All job control statements are set to invoke SQL/DS in single-partition mode.

  If you are running in:

  – Single-partition mode, update the EXEC PROC= statement to include your installation's procedure name for the SQL/DS database identification statements.
  – Multi-partition mode, delete the EXEC PROC= statement and change the EXEC statement for a multi-partition environment.
- You have to update the password in the EXEC statement.

To continue the DL/I EXTRACT DEFINEs process, perform the following steps:

1. Start the ISQL EXTRACT DEFINEs utility using the following EXTRACT parameter statement:

```
EXTRACT
   PCBNAME=pcbname,
   PSBNAME={psbname|(psbname,pcbnumber)},
   DLIPROC=procname,
   USERID=SQLDBA/password
   [,REPLACE]
```

2. Run the ISQL routine identified by the PCBNAME parameter specified in the above EXTRACT statement.
3. Complete the ISQL processing by:

   Creating the Target Tables

Issuing the EXTRACT command
Issuing the SUBMIT command

```
// JOB PREP DLZDLBD
// LIBDEF *,SEARCH=(SQL.sublib,DLIPRD.sublib,DLIGEN.sublib,...)
// LIBDEF PHASE,CATALOG=DLIPRD.sublib
// EXEC PROC=(identification statements for sql/ds database)
// DLBL IJSYSPH,'punch-work-file',0
// EXTENT SYSPCH,,1,0,xxx,yyy
// DLBL IJSYSIN,'punch-work-file',0
// EXTENT SYSIPT,,1,0,xxx,yyy
ASSGN SYSPCH,DISK,VOL=volid,SHR
* ------------------------------------------------------- *
* STEP1:                                                  *
*       PREPROCESS DLZDLBD                                *
* ------------------------------------------------------- *
// EXEC PGM=ARISQLDS,SIZE=AUTO,PARM='SYSMODE=S,PROGNAME=ARIPRPA/PREPNAME=DLZDLBDP,  *
       USERID=sqldba/sqldbapw'
READ MEMBER DLZDLBD
/+
/*
CLOSE SYSPCH,PUNCH
* ------------------------------------------------------- *
* STEP2:                                                  *
*       CATALOG PREPROCESSED SOURCE TO DLZDLBDP.SQL       *
*       NOTE: FOR LIBRARIAN PURPOSES THE SOURCE MUST NOT  *
*             BE OF TYPE "A"                              *
* ------------------------------------------------------- *
ASSGN SYSIPT,DISK,VOL=volid,SHR
// EXEC LIBR,PARM='ACCESS S=DLIGEN.sublib; CATALOG DLZDLBDP.SQL REPLACE=Y'
/*
CLOSE SYSIPT,SYSRDR
* ------------------------------------------------------- *
* STEP3:                                                  *
*       REMOVE OLD DLZDLBDP.A                             *
*       RENAME DLZDLBDP.SQL ----> DLZDLBDP.A              *
* ------------------------------------------------------- *
// EXEC LIBR,PARM='MSHP'
ACCESS S=DLIGEN.sublib
DEL DLZDLBDP.A
REN DLZDLBDP.SQL : DLZDLBDP.A
/*
* ------------------------------------------------------- *
* STEP4:                                                  *
*       ASSEMBLE  DLZDLBDP.A                              *
* ------------------------------------------------------- *
ASSGN SYSPCH,DISK,VOL=volid,SHR
// OPTION CATAL,DECK,ALIGN
// EXEC ASSEMBLY,SIZE=1024K
         COPY   DLZDLBDP
/*
CLOSE SYSPCH,PUNCH
* ------------------------------------------------------- *
```

```
* STEP5:                                                  *
*       CATALOG  DLZDLBDP.OBJ                             *
* ------------------------------------------------------- *
ASSGN SYSIPT,DISK,VOL=volid,SHR
// EXEC LIBR,PARM='ACCESS S=DLIPRD.sublib'
/*
CLOSE SYSIPT,SYSRDR
* ------------------------------------------------------- *
* STEP6:                                                  *
*       LINKEDIT DLZDLBDP.OBJ                             *
* ------------------------------------------------------- *
// OPTION CATAL
  INCLUDE DLZDLBDP
// EXEC LNKEDT
/*
/&
```

Figure 6. Sample Job PREP DLZDLBD

Chapter 6. Post Installation Tasks and Tools

Getting DL/I Sample Application Job Streams

To verify the correct installation of DL/I VSE 1.12.1, you can use the two sample application job streams that are described in "DL/I Sample Application" in the DL/I DOS/VS User's Guide.

1. The online sample application job stream is in the DL/I production sublibrary as member DLZSAMJS.A. Run the following sample job to get this member from the DL/I production sublibrary:

```
// JOB PUNCH
// EXEC LIBR
  ACCESS SUBLIB=DLIPRD.sublib
  PUNCH DLZSAMJS.A
/*
/&
```

2. The access sample application job stream is in the DL/I production sublibrary as member DLZSAMAC.A.

You can retrieve all other DL/I sample programs in the same way (refer to “Source Books in the Production Sublibrary” on page 12).

Creating an MSHP Retrace

After you have installed DL/I VSE 1.12.1, you might want to print the DL/I related information contained in the System History File.

1. *IBM Service* panel of the VSE Interactive Interface

   4 (Retrace History File), and then

   7 (Retrace Component ID)

2. Run a job as shown below:

```
// JOB RETRACE
// EXEC MSHP,SIZE=900K
     RETRACE COMP ID=5746-XX1-00-3IO
/*
/&
```

Displaying DL/I Library Directory Listings

To display the directory listings of your DL/I VSE 1.12.1 sublibraries, run the jobs in Figure 7 on page 27 and Figure 8 on page 28:

```
// JOB DSPLYPRD
// EXEC LIBR
LISTD SUBL=DLIPRD.sublib
/*
/&
```

Figure 7. Sample Job DSPLYPRD

```
// JOB DSPLYGEN
// EXEC LIBR
LISTD SUBL=DLIGEN.sublib
/*
/&
```

Figure 8. Sample Job DSPLYGEN

DL/I Link Book

If you change any of the DL/I source modules, the affected DL/I production sublibrary phases must be relinked. Member DLZLNKBK.A in the DL/I production sublibrary contains job control statements to link all DL/I modules. You can punch out this link edit book using the VSE librarian program with the PUNCH command. Then, read the comments, alter the statements that need to be adapted and run the job.

Linking certain DL/I phases may produce various linkage editor messages. For more information about these messages, refer to "Error Messages while Reassembling DL/I Components" on page 31.

DL/I Merge Book

If you want to merge any or all of the DL/I code from the DL/I libraries to any other libraries, you can use the merge book (DLZMERGE.A) that is included in the DL/I production sublibrary. You can punch out this merge book using the VSE librarian program with the PUNCH command. Then, read the comments, alter the statements that need to be adapted and run the job.

DL/I Delete Book

If you want to delete any or all DL/I code from the libraries where your DL/I code resides, you can use the delete book (DLZDLETE.A) that is provided in the DL/I production sublibrary. You can punch out this delete book using the VSE librarian program with the PUNCH command. Then, read the comments, alter the statements that need to be adapted and run the job.

The delete book also removes the entry of DL/I in the VSE System History File.

Debugging Tools

To assist in problem determination and solution, DL/I produces three types of dumps:

Unformatted Dumps
Display register contents and the contents of a section of storage at the time of the failure.

Formatted Dumps
Provide additional information by means of:

- Locating DL/I control blocks in storage
- Dumping each control block separately
- Identifying each block with a control block heading

Problem Determination Dumps
Are unformatted dumps that are identified by a special header and written to a special dump data set.

DL/I dumps are taken in the event of the following:

- Online task failure
- Online system failure
- Batch application failure
- MPS application failure
- Batch utility failure (where the utility is using DL/I)

where failure is an abnormal ending of a job or task.

The following table shows what type of dump is useful when analyzing one of the above failures:

Type of Failure	**Unformatted Dump**	**Formatted Task Dump**	**Formatted System Dump**	**Problem Determination Dump**
Online Task		x		
Online System			x	x
Batch Application			x[1]	x
MPS application	x			x
Batch utility	x		x	x
[1] Only if `//OPTION NOSYSDMP` specified in JCL.				

For additional information about DL/I debugging facilities and dump selection, refer to "Interpreting And Debugging Dumps" in the DL/I DOS/VS Diagnostic Guide.

Chapter 7. Applying Service

Note: Tape delivery of PTFs for z/VSE has been discontinued. PTFs are available on DVD or from the Internet.

Service to DL/I modules is usually provided in the form of a PTF (Program Temporary Fix). To apply a PTF, use the following dialogue path starting from the main menu of the VSE Interactive Interface:

1
Installation

4
IBM Service

2
PTF Handling

Sometimes you can also apply service by means of an APAR (Authorized Program Analysis Report) fix. In this case you have to run the update job supplied with the APAR documentation or distributed through the IBM support center. The update job re-assembles and re-links the module(s) to be serviced.

Error Messages while Reassembling DL/I Components

If you intend to assemble any of the DL/I module source books from the DL/I libraries, certain assembly error or warning messages may result. Such messages do not affect the assembler output in any way. They can be ignored and the assembly has completed successfully. Described below are the assembler messages you may encounter:

- ASMA033I Storage alignment for *label* unfavorable

  This warning message may be issued during the assembly of the DL/I modules:

  ```
  DLZTPRT0 and DLZTRPR0.
  ```

- ASMA300W USING overridden by a prior active USING on statement number *nn*.

  This warning message may be issued during the assembly of the following DL/I modules:

  ```
  DLZBACM0, DLZCUMM0, DLZDHDS0, DLZDLBL0, DLZDLBL1, DLZDLOC0, DLZDLRA0,
  DLZDLRB0, DLZDLRC0, DLZDLRD0, DLZDLRE0, DLZDLRF0, DLZDMPM0, DLZDXMT0,
  DLZLGPM0, DLZRDBC0, DLZRDBM0, DLZRRC00, DLZTXIT0.
  ```

- ASMA301W Prior active USING on statement number *nn* overridden by this USING

  This warning message may be issued during the assembly of the following DL/I modules:

  ```
  DLZDDLE0, DLZDLA01, DLZDLBL0, DLZDLBL1, DLZDLBL2, DLZDLBL3, DLZDLD00,
  DLZDLOC0, DLZDLTXX, DLZDSEH0, DLZDXMT0, DLZFSDP0, DLZFTDP0, DLZLOC00,
  DLZLOGP0, DLZMPI00, DLZODP,   DLZUDMP0, DLZURDB0, DLZURGL0, DLZURGS0,
  DLZURGU0, DLZURG10.
  ```

- ASMA302W USING specifies register 0 with a non-zero absolute or relocatable base address

  This warning message may be issued during the assembly of the following DL/I modules:

  ```
  DLZDLRA0, DLZDLRB0, DLZDLRC0, DLZDLRD0, DLZDLRE0, DLZDLRF0, DLZDLRG0.
  ```

- ASMA303W Multiple address resolutions may result from this USING and the USING on statement number *nn*

  This warning message may be issued during the assembly of the following DL/I modules:

  ```
  DLZBACK0, DLZBNUC0, DLZCBDP0, DLZDBH00, DLZDHDS0, DLZDLA00, DLZDLBDP,
  DLZDLBL1, DLZDLBL2, DLZDLBL3, DLZDLBPP, DLZDLD00, DLZDLRA0, DLZDLRB0,
  DLZDLRC0, DLZDLRD0, DLZDLRE0, DLZDLRF0, DLZDLRG0, DLZDLTXX, DLZDPSB0,
  ```

```
DLZDSEH0, DLZDXMT0, DLZEIPB1, DLZEIPO0, DLZEXDFP, DLZLBLM0, DLZLOGP0,
DLZMPI00, DLZODP,   DLZOLI00, DLZPRPAR, DLZPRURC, DLZQUEF0, DLZRDBL0,
DLZRDBL1, DLZRDBM0, DLZRRC00, DLZUACB0, DLZUCUM0, DLZUDMP0, DLZURDB0,
DLZURGL0, DLZURGP0, DLZURGS0, DLZURGU0, DLZURG10, DLZURRL0, DLZURUL0,
DLZTPRT0.
```

Error Messages while Relinking DL/I Components

When relinking certain DL/I modules and creating DL/I phases for the production sublibrary, certain warning messages may appear on the console and in the output (SYSLST) of the link edit job. In most of the cases the link edit process continues successfully. Described below are some of the warning link edit messages you may encounter on your console log and/or on SYSLST (output listing):

- 2158I NO CSECT LENGTH SUPPLIED

  The END statement does not contain the length of the control section. Allow the job to execute. If execution fails, reassemble (recompile) and relink the module in question. If execution is successful, ignore the message. This may appear when link editing phases such as DLZMPI00, DLZRRC00, DLZBPC00.

- 2199I ERROR HAS OCCURRED DURING LINKAGE EDITING

  The specific messages will appear on SYSLST and processing will continue. Review the messages, take any action specified and continue running the link edit job.

- 2139I DUPLICATE SECTION DEFINITION _____ **** SECTION IGNORED ****

  This may appear when link editing a number of DL/I phases and the associated modules and CSECTs included. The link edit should continue to process, then review the output (SYSLST) for any additional information or action required.

  Listed below are the DL/I phases where this condition can occur:

  ```
  DLZBACK0,  DLZBPC00,  DLZBNUC0,  DLZDBH00,  DLZDHDS0,
  DLZDLA00,  DLZDLBDP,  DLZDLBPP,  DLZDLR00,  DLZDSEH0,
  DLZEXDFP,  DLZFSDP0,  DLZLOGP0,  DLZMPI00,  DLZMPX00,
  DLZRDBL0,  DLZRRC00,  DLZTPRT0,  DLZUACB0,  DLZUCUM0,
  DLZUDMP0,  DLZURDB0,  DLZURGL0,  DLZURGP0,  DLZURGS0,
  DLZURGU0,  DLZURG10,  DLZURPR0,  DLZURRL0,  DLZURUL0.
  ```

  In addition the following CSECTs (duplicate section names) may also appear in the above message when link editing certain DL/I phases:

  ```
  DLZCONSL,  DLZDIMOD,  DLZLGPCN,  DLZLGPMT,  DLZRDR,
  DLZRRC10,  DLZTPCN,   IJJFCBIC,  OPENWORK,  SCDCSECT.
  ```

- CONTROL SECTIONS OF ZERO LENGTH IN INPUT

  This may appear when link editing such phases as:

  ```
  DLZBNUC0,  DLZDDLE0,  DLZDHDS0,  DLZDLRA0,  DLZMDLI0,
  DLZMPI00,  DLZMPUR0,  DLZMSTP0,  DLZMSTR0,  DLZOLI00,
  DLZRRC00,  DLZUACB0.
  ```

  The link edit should continue and you may ignore the message and continue processing.

- UNRESOLVED EXTERNAL REFERENCES

  This may appear when link editing phases DLZDLBDP, DLZDLBPP, and DLZEXDFP. The link edit should continue to execute and the message may be ignored.

- POSSIBLE INVALID ENTRY POINT DUPLICATION IN INPUT

  This may occur in some phases such as DLZDHDS0 and DLZUDMP0. The link edit should continue processing and you may ignore the message.

- UNRESOLVED ADDRESS CONSTANTS

  This is a valid condition when link editing a number of DL/I phases such as:

- DLZURG10, DLZBACK0 (with CICS), DLZDBH00.
- DLZACT (application control table, which is the method of link editing the DL/I online nucleus: DLZODP, DLZNUC*xx*, and DLZNUC).
- DLZRDBC0 (when linked with DFHDBP and DFHDLBP).

The condition also occurs when link editing the modules DLZDLBDP, DLZDLBPP, and DLZEXDFP.

The unresolved external references are caused by weak external references (WXTRNs) which cause no operational error. Check the output (SYSLST) and continue the link edit processing.

Chapter 8. Features Introduced with Earlier Releases of DL/I

Features Introduced with DL/I DOS/VS 1.7.1

The features are available for all users of DL/I 1.7.1 and DL/I 1.7.0 users with SPE PTF UL90011:

- With VSE/Advanced Functions 2.1 (part of VSE/SP 2.1 and VSE/SP 3.1), DL/I can make full use of the VSE Virtual Addressability Extension with Multiple Partition Support (MPS).
- The new GSTA call can be used to acquire system statistics.

Improved Multiple Partition Support

DL/I supports separate address spaces for MPS partitions. However, CICS and DL/I must still be in the same partition. With the exception of applications using GSCD to get buffer statistics or to reference other DL/I control blocks, the support is transparent. No special procedures are required by DL/I for initialization of a DL/I MPS environment in multiple address spaces.

Dynamically Scheduling MPS or Non-MPS Execution

If you want to run a DL/I batch job in an MPS batch partition and you are not sure if MPS is already started, you can use the dynamic initialization program DLZCTRL. DLZCTRL will fetch either:

DLZRRC00 (if MPS is not active), or
DLZMPI00 (if MPS is already started).

By specifying DLZCTRL on the EXEC statement, you can dynamically schedule MPS or non-MPS execution. No other changes in the job control statements are required.

Note: The function has been extended with DL/I 1.11 as described in “Multiple MPS Systems” on page 47 and through APAR PQ54003. Refer to “DL/I VSE 1.11 Enhancements” on page 61 for details.

The New GSTA Call

The new GSTA call (get statistics call) returns to the application a block containing statistics related to system operations and to the application from which the call was issued.

Applications referencing DL/I control blocks and control areas via GSCD must operate in the same address space as DL/I. Applications using GSCD for statistics retrieval that you want to run in a VSE address space other than that of DL/I will have to be modified to use the new GSTA call function.

GSTA is valid from all DL/I environments, except for the following:

- GSTA is not supported for the HLPI interface.
- GSTA is not supported for the RPG interface.
- GSTA is not supported for remote PSBs (but GSTA is supported for extended remote PSBs).

GSTA Call Format

You can issue the GSTA call in assembler language, COBOL, or PL/I application programs, by using the standard DL/I call format with the following six parameters:

1. "5" as parameter count (parmcnt).
2. "GSTA" as the function code.
3. A valid PCB.

4. The statistics buffer.
5. The statistics buffer length (bufflen).
6. A return code field.

For an example of how to use the GSTA call in an assembler application program, refer to Figure 9 on page 36 .

```
PGMSTART     ...                Start of your application program
               ...
             MVC   GSTAPCB,0(1)   Get PCB address for GSTA call
             USING STBF,6         Establish addressability for DLZSTBF
             LA    6,BUFFER
               ...
             LA    1,GSTAPARM     Load GSTA call parameter list
             CALL  ASMTDLI        DL/I GSTA call
             L     15,GSTARCOD
             LTR   15,15          Check return code
             BNZ   GSTAERR        GSTA call not successful
               ...
GSTAERR      ...
             ...
*-----------------*
*  constant area  *
*-----------------*
               ...
GSTAPARM   DC    A(PARMCNT)       Parameter-count address
           DC    A(GSTAFUNC)      Call function address
GSTAPCB    DC    A(0)             Address of valid PCB
           DC    A(BUFFER)        Buffer start address
           DC    A(BUFFLEN)       Buffer length address
           DC    A(GSTARCOD)      Return code address
PARMCNT    DC    F'5'             Parameter count: must be 5
GSTAFUNC   DC    CL4'GSTA'        Call function: must be GSTA
BUFFER     DS    CL108            Statistics buffer
BUFFLEN    DC    F'108'           Length of statistics buffer
GSTARCOD   DC    F'0'             Four-byte area for return code
           DLZSTBF                Macro call for DLZSTBF
             ...
           END                    End of program
```

Figure 9. How to Invoke the New GSTA Call in an Assembler Program

GSTA Return Codes

Upon return, GSTARCOD contains the four-byte return code *rc*:

Table 4. GSTA Call Return Codes

rc dec	rc (hex)	Explanation and Returned Data
0	(0000)	The GSTA call was successful. DL/I returns the full statistics.
4	(0004)	The passed buffer length (bufflen) is less than the statistics buffer length (108 bytes). DL/I returns the first bufflen bytes of the statistics only.
8	(0008)	The passed buffer length (bufflen) is more than the statistics buffer length (108 bytes). DL/I returns the statistics to the low-address part of the buffer.
12	(000C)	The PSB I/O work area is too small. This usually means that an ACBGEN run is required for the application PSB. No data is returned.
16	(0010)	The passed buffer length (bufflen) is not positive or is greater than 32K. No data is returned.
20	(0014)	The number of specified parameters (parmcnt) is greater than 5. No data is returned.

Table 4. GSTA Call Return Codes (continued)		
rc dec	**rc (hex)**	**Explanation and Returned Data**
24	(0018)	The application runs with remote PSB. No data is returned.

Layout of the Statistics Buffer

With the DL/I macro DLZSTBF, the returned statistics information can be referenced:

```
DLZSTBF   DSECT
STBF      DS   0F       Statistics buffer
*
STBFS     DS   0F       System statistics
STBFTSKC  DS   PL8      Total number of PSB scheduling calls
STBFDLOC  DS   PL4      Total number of program isolation
*                       deadlock occurrences
STBFCMTC  DS   PL4      Number of times at current max task
STBFPDUP  DS   PL4      Number of duplicate PSBs created
STBFRQCT  DS   F        Number of requests received
*                       by the buffer handler
STBFINPL  DS   F        Number of requests satisfied
*                       from buffer pool
STBFRDST  DS   F        Number of read requests issued
STBFALTR  DS   F        Number of buffer alter requests received
STBFOSWT  DS   F        Number of writes issued
STBFBKWT  DS   F        Number of blocks written
STBFNWBK  DS   F        Number of new blocks created in pool
STBFCHWT  DS   F        Number of chained writes issued
STBFCHBK  DS   F        Number of blocks written on write chain
STBFISTL  DS   F        Number of retrieve by key calls
STBFIGET  DS   F        Number of GN calls for KSDS received
STBFWERR  DS   X        Number of permanent write error buffers
*                       in the pool
STBFWERT  DS   X        Largest number of write error buffers
*                       ever in the pool
          DS   H        Reserved
STBFSEND  DS   0F       End of system statistics
*
STBFJ     DS   0F       Job statistics
STBFGU    DS   F        Number of GU   calls issued
STBFGN    DS   F        Number of GN   calls issued
STBFGNP   DS   F        Number of GNP  calls issued
STBFGHU   DS   F        Number of GHU  calls issued
STBFGHN   DS   F        Number of GHN  calls issued
STBFGHNP  DS   F        Number of GHNP calls issued
STBFISRT  DS   F        Number of ISRT calls issued
STBFDLET  DS   F        Number of DLET calls issued
STBFREPL  DS   F        Number of REPL calls issued
STBFCHKP  DS   F        Number of CHKP calls issued
STBFJEND  DS   0F       End of job related statistics
STBFEND   DS   0F       End of buffer
*
STBFLEN   EQU  STBFEND-STBF     Length of statistics buffer
STBFSLEN  EQU  STBFSEND-STBFS   Length of system statistics
STBFJLEN  EQU  STBFJEND-STBFJ   Length of job statistics
```

Figure 10. Layout of the Statistics Buffer DLZSTBF

Features Introduced with DL/I DOS/VS 1.8

Date and Time on Reports and Statistics

Reports and statistics provided by several DL/I utilities are date and time stamped and have headings of a unique form.

Literal String in the HLPI WHERE Clause

The DL/I High Level Programming Interface (HLPI) is an easy-to-use method for processing DL/I databases. It provides commands similar in syntax to those in CICS command language.

The existing syntax has been extended to allow the use of literals with the WHERE option in PL/I programs. With this support, the WHERE option will be compatible with the OS EXEC DLI language.

The method for selecting a segment with the WHERE clause using HLPI for PL/I programs is as follows:

```
WHERE(name op value)
```

where:

- *name* is the name of any field defined in the segment.
- *op* is a relational operator.
- *value* is a comparative value, which is:
  - a variable name declared in the host language of the application program, or
  - a character string for PL/I programs. The literal must be enclosed in quotes.

The HLPI support is dependent on the CICS EXEC translator support.

Within syntax checking by the CICS EXEC translator, the following errors are detected by the translator:

- Invalid text in the WHERE clause
- Missing quote

Diagnostic messages issued by the CICS EXEC translator are described in the VSE Messages and Codes publications.

Conditional Job Control Support

DL/I supports the use of conditional JCL for VSE. The DL/I batch initialization program DLZRRC00, the MPS batch initialization program DLZMPI00, and the DL/I utilities always provide return codes. Return codes provided by user-written programs are passed on by request.

For details, refer to *DL/I Resource Definition and Utilities*.

Status Code for Read-Only Programs

A new processing option, PROCOPT, is provided for read-only programs to give the application a possibility to react on invalid pointer situations and to reduce the number of abnormal terminations.

For details refer to *DL/I Resource Definition and Utilities*.

Partial Data Base Load

This feature allows initial loading of a database with a large amount of data in more than one step, using PROCOPT=L. Read-only access within the several load steps is possible.

For details refer to *DL/I Resource Definition and Utilities*.

Increased Maximum Data Set Control Interval Size

DL/I 1.8 supports VSAM CI-sizes of its databases up to a maximum of 30KB.

For details refer to *DL/I Resource Definition and Utilities*.

Performance Improvements for Data Set Image Copy/Recovery

A block size for DL/I backup tapes can optionally be specified.

For details refer to *DL/I Resource Definition and Utilities*.

Device Independence

This feature is available for all users of DL/I 1.8 with APAR PL48345.

Due to a new CKD specification, DL/I database definition is device-independent for CKD devices. If you use the DATASET parameter during the database definition, you may code a new specification of the DEVICE parameter:

DEVICE=CKD Must only be specified for CKD devices.
DEVICE=FBA Must only be specified for FBA devices.

Coding Example:

```
DATASET DEVICE=CKD ...
```

The CKD device type specifications (for example, 3380) need no longer be used but are still supported. The new device specification is also supported by the Interactive Macro Facility (IMF).

For new messages refer to Chapter 10, “Messages and Codes,” on page 93 .

Automatic Verification for DL/I ESDS Data sets

This feature is available for all users of DL/I 1.8 with APAR PL20988.

Automatic Verification for DL/I ESDS data sets allows you to have just one CICS-DL/I startup with no VERIFY-statements in it, independent of a normal or an emergency start. VSE/VSAM 1.3 provides automatic verification for all VSAM data sets doing normal record processing. Thus, the index component of a DL/I database will be verified automatically. Automatic verification does now also apply to a data component of a DL/I database (ESDS) working in Control Interval Processing Mode (CNV).

Program Isolation Enhancement

This feature is available for all users of DL/I 1.8 with APAR PL55587.

Due to the DL/I program isolation enhancement, a long running transaction may no longer cause a large amount of storage to be acquired and not freed.

Features Introduced with DL/I DOS/VS 1.9

DL/I 1.9 supports the VSE/ESA dynamic partitions, an enhanced database I/O error handling and the CICS/VSE 2.1 XRF function.

Dynamic Partition Support

DL/I runs in static partitions as before. DL/I batch and MPS jobs can also run in VSE/ESA dynamic partitions, but all restrictions for dynamic partitions apply. For DL/I, this means:

- MPS restart is not possible, because Checkpoint/Restart is not supported in a dynamic partition.
- The following utilities may run in dynamic partitions:
  - ACBGEN (DLZUACB0), and
  - Data Base Pre-Reorganization (DLZURPR0),

  but the system logical units SYSRDR, SYSIPT, SYSPCH, and SYSLST may not be assigned to a disk.

  For example, you may not run a job in a dynamic partition if you

  assign SYSPCH to a disk in the first job step, and then
  reassign the disk to SYSIN in a following job step to process the output of the first job step.

For a detailed description of VSE/ESA dynamic partitions and applicable restrictions, refer to z/VSE Planning.

Database I/O Error Handling

With APARs PM56161 and PH29950, new DL/I recovery capabilities are available. They have been added to the description formerly presented here. The new, updated description is shown in Chapter 9, “Recovery after a DLZ004I or DLZ005I Error,” on page 69.

DL/I in a CICS/XRF Environment

VSE provides the CICS/VSE 2.1 Extended Recovery Facility (CICS XRF). The facility guarantees high availability by an automatic system recovery after a failure. For a detailed description of CICS XRF refer to the CICS/VSE XRF Guide.

Within a DL/I CICS XRF environment, there is an active and an alternate DL/I CICS system:

- The active DL/I CICS system performs the normal DL/I processing.
- The alternate CICS waits in a standby mode until the active CICS terminates due to an error situation or a console command, and then takes over the DL/I CICS system including the DL/I databases.

The DL/I system of the alternate CICS is partially initialized because the DL/I databases can only be owned either by the active CICS or the alternate CICS. During takeover, the alternate CICS completes the DL/I initialization, opens the DL/I databases and performs a backout if required. If in the active DL/I CICS system an I/O error occurs with a DL/I database, the user is asked during takeover whether the DL/I initialization is to be continued or canceled.

Refer also to “Operating in a CICS XRF Environment” on page 72.

Requirements

- Shared Databases

  For DL/I data sets, specify DISP=SHR. For more information about sharing data sets, see the CICS/VSE XRF Guide.

- Shared CICS System Log

  A DL/I CICS XRF environment requires CICS system journals shared by the active CICS and the alternate CICS. The CICS system journal must be defined with DISP=SHR in JCL. For details, see the CICS/VSE XRF Guide.

  If the DL/I CICS XRF system runs without a CICS system journal, DL/I initialization is stopped with message DLZ140I.

- Compatible definitions for active and alternate DL/I CICS

  The active and the alternate DL/I CICS systems must have compatible DL/I and CICS resource definitions. There will not be any consistency check for DL/I or CICS resources such as CICS tables, DL/I nucleus (ACT, DBD, PSB, and so on).

Setup

To set up a DL/I CICS XRF environment, the SIT parameters XRF and DLIOER must be coded as follows:

```
XRF=YES
DLIOER=ABEND|CONTINUE
```

For a detailed description on how to set up a CICS XRF environment, see the CICS/VSE XRF Guide.

System Integrity

Depending on the online DL/I system configuration (logging specified), data integrity is ensured because alternate CICS together with DL/I performs backout during takeover processing. This applies also for DL/I MPS batch tasks and CICS ISC environments.

Until you repaired the database that has been stopped by an I/O error, automatic backout during CICS restart (EMER/TAKEOVER) will be suppressed for databases associated with any PSB that refers to that database. In this case, you have to perform a recovery of the damaged database as described in Chapter 9, “Recovery after a DLZ004I or DLZ005I Error,” on page 69.

Layout of Database Open Repaired Log Record (log id X'2F')

```
DLPOPEN  DSECT
DLOLEN   DS    H                 LENGTH OF OPEN RECORD
DLOSPACE DS    H'0'              ZERO
DLOCODE  DS    X                 RECORD TYPE CODE - X'2F'
         DS    X                 RESERVED
DLOORG   DS    X                 DATA SET ORGANIZATION
DLOESDS  EQU   X'00'             ESDS ORGANIZATION
DLOKSDS  EQU   X'04'             KSDS ORGANIZATION
DLOTORF  DS    CL1               TYPE OF OPEN RECORD FLAG
DLOIOERR EQU   X'80'             OPEN REPAIRED RECORD
         DS    XL4               BINARY ZERO
DLOCI    DS    XL2               CONTROL INTERVAL LENGTH
         DS    XL2               BINARY ZERO
DLOFILE  DS    CL8               DATA SET FILENAME (ACB)
DLODMB   DS    CL8               DMB NAME
DLOACBNO DS    CL1               DSG ACB NUMBER
DLOHESDS EQU   2                 HISAM ESDS
DLODATE  DS    CL3               BINARY ZERO
DLOTIME  DS    CL4               BINARY ZERO
DLOSEQNO DS    CL4               LOG RECORD SEQUENCE NUMBER
DLOEND   EQU   *
DLPOLEN  EQU   *-DLPOPEN         LENGTH OF OPEN LOG RECORD
DLOSEQ   DS    CL4               LOG RECORD SEQUENCE NUMBER
DLPOLENG EQU   *-DLPOPEN         LENGTH OF OPEN RECORD FOR
*                                LOG PRINT ROUTINE
```

Layout of Database I/O Error Stop Log Record (log id X'31')

```
DLPIOERR DSECT
DLILEN   DS    H                 LENGTH OF STOP RECORD
DLISPACE DS    H'0'              ZERO
DLICODE  DS    X                 RECORD TYPE CODE - X'31'
DLISTOP  EQU   X'31'             STOP LOG RECORD
         DS    X                 RESERVED
DLIORG   DS    X                 DATA SET ORGANIZATION
DLIESDS  EQU   X'00'             ESDS ORGANIZATION
DLIKSDS  EQU   X'04'             KSDS ORGANIZATION
DLISTLRF DS    X                 STOP LOG RECORD FLAG
DLIIOERR EQU   X'80'             STOPPED-IOERROR-RECORD
DLIPSB   DS    CL8               PSB NAME
DLIFILE  DS    CL8               DATA SET FILE NAME (ACB)
DLIDMB   DS    CL8               DMB NAME
DLIACBNO DS    CL1               DSG ACB NUMBER
DLIHESDS EQU   2                 HISAM ESDS
DLIDATE  DS    CL3               BINARY ZERO
DLITIME  DS    CL4               BINARY ZERO
DLISEQNO DS    CL4               LOG RECORD SEQUENCE NUMBER
DLIEND   EQU   *
DLPILEN  EQU   *-DLPIOERR        LENGTH OF STOP LOG RECORD
DLISEQ   DS    CL4               LOG RECORD SEQUENCE NUMBER
DLPILENG EQU   *-DLPIOERR        LENGTH OF STOP LOG RECORD
*                                FOR LOG PRINT ROUTINE
```

The DL/I Logprint Utility (DLZLOGP0)

The DL/I Logprint Utility (DLZLOGP0) is enhanced to print out new log records. The printout of an open repaired record is as follows:

```
OPEN REPAIRED RECORD, DSORG = (E)SDS (/HISAM)
                              (K)
SEQNO = xxxxxxxx, DMBNAME = aaaaaaaa, FILENAME = aaaaaaaa, REPAIRED
```

The printout of an I/O error stop record is as follows:

```
I/O ERROR STOP RECORD, DSORG = (E)SDS (/HISAM)
                                (K)
SEQNO = xxxxxxxx, DMBNAME = aaaaaaaa, FILENAME = aaaaaaaa, PSBNAME = aaaaaaaa
```

where:

```
xxxxxxxx = hexadecimal numbers
aaaaaaaa = alphanumeric characters
```

Note: When specifying LO / LS control statements, remember that PSBname and DBDname are part of these log records.

Features Introduced with DL/I DOS/VS 1.10

The following enhancements are available with DL/I DOS/VS 1.10:

- DL/I Applications above the 16 MB line of Storage
- VSE/VSAM HS-Buffers above the 16 MB Line of Storage
- Virtual Disk Exploitation
- DL/I Run Statistics (DLZSTTL) Enhancements

Notes:

1. If you use the MPS Restart facility, your partition must not be larger than 16 MB.
2. When using the CICS/VSE 2.2 distributed program link (DPL) function, the following DL/I commands are restricted in their use for application programs running in the server region:

   ```
   EXEC DLI TERM
   CALL DL/I TERM
   EXEC DLI CHKP
   CALL DL/I CHKP
   ```

   For more information on DPL, refer to the *CICS/VSE 2.2 Release Guide*.
3. With CICS/VSE 2.2, the name of the CICS intercommunication mirror program has been changed from DFHMIR to DFHMIRS. However, DL/I 1.10 still accepts the old name DFHMIR in the DLZACT TYPE=PROGRAM statement, but creates an entry for DFHMIRS in the Application Control Table (ACT) phase. To avoid an MNOTE in the ACT generation, specify DFHMIRS in the DLZACT TYPE=PROGRAM statement.

   Using REMOTE=YES in the DLZACT TYPE=CONFIG statement generates the equivalent of a DLZACT TYPE=PROGRAM,PGMNAME=DFHMIRS statement, which includes the PSB names of all PSBs that are in the DL/I online nucleus.

DL/I Applications above the 16 MB Line of Storage

Note: This support has been further enhanced. Refer to "Support for DL/I 31-Bit Applications (PN67649)" on page 53 for a complete description and the latest information.

VSE/VSAM HS-Buffers Above the 16 MB Line of Storage

DL/I exploits the VSE/VSAM support that allows you to allocate input/output buffers above the 16MB line of storage. Specifically, VSE/VSAM input/output buffers may be allocated above the 16MB line of storage for the use of DL/I:

- KSDS index files
- HISAM KSDS and ESDS data files
- SHISAM KSDS data files.

In the following, these buffers are referred to as HS-buffers.

Up to and including DL/I 1.9, you were able to control the allocation of HS-buffers by specifying the number of buffers in the **HSBFR** parameter, or by using a default. With the new support in DL/I 1.10, you can also specify the residence of these buffers.

Depending on the number of HS-buffers defined above the 16MB line of storage, the VSE/VSAM input/output operations are reduced for the above listed files. The CPU time overhead from moving data due to the change from VSE/VSAM LOCATE mode to VSE/VSAM MOVE mode can be ignored.

Considerations

- The buffer pool management for HDAM/HIDAM ESDS files (VSAM user buffering) is not changed. That is, buffer allocation beyond 16 MB is not allowed.
- The new parameters HSMODE=ANY|BELOW are also reflected in the DL/I interactive functions, IMF and IUG.

  For information on IMF and IUG, refer to "DL/I IMF and IUG Functions" on page 21, and the publications:

  – DL/I DOS/VS User's Guide
  – *DL/I Interactive Resource Definition and Utilities*

Compatibility of Existing Applications

The DL/I support for allocating HS-buffers above the 16MB line of storage does not affect your existing DL/I applications. That is, if the parameter HSMODE is not specified (this is the case in all existing DL/I batch jobs and online ACT generations), DL/I assumes the default HSMODE=BELOW. This places the HS-buffers below the 16 MB line of storage (as was the case prior to DL/I 1.10).

On the other hand, if you want the HS-buffers to reside above the 16 MB line of storage, you have to specify buffer residence as described under "Invocation" on page 43.

Invocation

To specify buffer residence, use the new parameter **HSMODE** for the:

- **DL/I batch environment** in the DL/I control statement.

  The HSMODE parameter can be specified whenever the HSBFR parameter can be specified.

  For example, to allocate 50 KSDS index and data buffers in 31-bit address space for a batch program, a new DL/I batch control statement could be:

```
DLI,SAMPLPRG,PSB1,,HDBFR=(10),HSBFR=(50,50,,INDEX),HSMODE=ANY
```

- **DL/I online environment** during online nucleus generation in the DLZACT TYPE=CONFIG macro statement.

  For example, to allocate 50 KSDS index and data buffers in 31-bit address space for an online program, you have to provide two statements as follows:

  1. A DLZACT TYPE=CONFIG statement to specify the residence of the HS-buffers in 31-bit address space:

```
DLZACT TYPE=CONFIG,
     REMOTE=NO,
     BFRPOOL=3,
     HSMODE=ANY,
     MAXTASK=10,
     CMAXTSK=5,
     SLC=DLZSLC01,
     PI=YES
```

2. A DLZACT TYPE=BUFFER statement to specify the *number* of the HS-buffers:

```
DLZACT TYPE=BUFFER,
     HDBFR=(20,HIDAM2),
     HSBFR=(50,50,,INDEX2)
```

An equivalent statement is required for each individual data base.

Applicable for both environments:

HSMODE=ANY
first tries to allocate buffers above the 16MB line of storage. If the attempt fails (all storage above the 16MB line of storage consumed or partition too small), HSMODE=ANY tries to allocate storage below the 16MB line of storage.

HSMODE=BELOW (the default)
specifies that buffers are to be located below the 16MB line of storage.

Virtual Disk Exploitation

All DL/I utilities can now allocate their work files on a virtual disk (that is, in virtual data space). The utility jobs do not have to be changed, however the job control statements (ASSGN, DLBL and EXTENT) must address the virtual disk instead of the real device. Allocating work files on a virtual disk significantly improves performance by reducing input/output operations to a real device.

The following DL/I utilities can benefit from virtual disk utilization:

- Initial Load
- Partial Load
- DLZURGL0 HD Reorganization Reload
- DLZURPR0 Data Base Pre-reorganization
- DLZURGS0 Data Base Scan
- DLZURG10 Data Base Prefix Resolution
- DLZURGP0 Data Base Prefix Update
- DLZPRCTn Partial Data Base Reorganization
- DLZUCUM0 Data Base Change Accumulation

Because virtual disks are not permanent, they should only be used for files that can be recovered in case of loss (for example, after IPL).

For more information on virtual disk support, refer to z/VSE Extended Addressability.

DL/I Run Statistics (DLZSTTL) Enhancements

The program DLZSTTL has been enhanced to provide new and improved information about the database activities. This information can assist the database administrator to optimize the DL/I database allocation.

The changes are as follows:

- New summary of DL/I calls. Calls are accumulated from DL/I startup to the execution of DLZSTTL.
- Updated buffer pool call summary.
- Important information from the buffer pool statistics are shown per subpool.
- New statistics per database. VSE/VSAM information is shown for non-HD databases.

Figure 11 on page 45 shows an example of the output of the DL/I Run Statistics program DLZSTTL.

```
DLZSTTL ============================================================
DLZSTTL     DL/I  DATA BASE- AND SUBPOOL-STATISTICS    1999/094    9:23:33
DLZSTTL ============================================================
DLZSTTL     DL/I RUN STATISTICS
DLZSTTL ------------------------------------------------------
DLZSTTL - NUMBER OF PSB SCHEDULING CALLS                     6
DLZSTTL - NUMBER OF TIMES AT PI DEADLOCK                     -
DLZSTTL - NUMBER OF TIMES IN PI WAIT                         4
DLZSTTL - NUMBER OF TIMES AT CURRENT MAX TASK                -
DLZSTTL - NUMBER OF DUPLICATE PSBS CREATED                   -
DLZSTTL ------------------------------------------------------
DLZSTTL - NUMBER OF GU   CALLS ISSUED                        -
DLZSTTL - NUMBER OF GN   CALLS ISSUED                        1
DLZSTTL - NUMBER OF GNP  CALLS ISSUED                        -
DLZSTTL - NUMBER OF GHU  CALLS ISSUED                        2
DLZSTTL - NUMBER OF GHN  CALLS ISSUED                      120
DLZSTTL - NUMBER OF GHNP CALLS ISSUED                        -
DLZSTTL - NUMBER OF ISRT CALLS ISSUED                       11
DLZSTTL - NUMBER OF DLET CALLS ISSUED                        4
DLZSTTL - NUMBER OF REPL CALLS ISSUED                        3
DLZSTTL - NUMBER OF CHKP CALLS ISSUED                        -
DLZSTTL ------------------------------------------------------
DLZSTTL     DL/I BUFFER POOL STATISTICS
DLZSTTL ------------------------------------------------------
DLZSTTL - NUMBER OF BUFFER HANDLER REQUESTS                1221
DLZSTTL - NUMBER OF REQUESTS SATISFIED FROM POOL            770
DLZSTTL - NUMBER OF BUFFER READ REQUESTS ISSUED             309
DLZSTTL - NUMBER OF BUFFER ALTER REQUESTS ISSUED             21
DLZSTTL - NUMBER OF BUFFER WRITE REQUESTS ISSUED             11
DLZSTTL - NUMBER OF NEW BLOCKS CREATED IN POOL               -
DLZSTTL - NUMBER OF CHAINED WRITE REQUESTS ISSUED            -
DLZSTTL - NUMBER OF BLOCKS WRITTEN ON WRITE CHAIN            -
DLZSTTL - NUMBER OF RETRIEVALS BY KEYED CALLS               81
DLZSTTL - NUMBER OF RETRIEVALS BY GETNEXT CALLS             46
DLZSTTL - NUMBER OF RETRIEVALS BY RBA CALLS                  -
DLZSTTL - NUMBER OF PERMANENT WRITE ERRORS                   -
DLZSTTL - NUMBER OF BUFFER PURGE CALLS                       5
```

Figure 11. Example: Output of the DL/I Run Statistics Program (DLZSTTL)

```
DLZSTTL ================================================================
DLZSTTL     DL/I COMMON SUBPOOL AND DATA BASE INFORMATION
DLZSTTL ================================================================
DLZSTTL - NUMBER OF SUBPOOLS                                  4
DLZSTTL
DLZSTTL -  DL/I RELATED INFORMATION PER SUBPOOL -
DLZSTTL
DLZSTTL SUB-     NUMBER  NUMBER OF    BUFFER      NUMBER OF  REQ. SATISFD     NUMBER OF     NUMBER OF
NUMBER OF
DLZSTTL POOL    OF DMBS    BUFFERS      SIZE    BH REQUESTS     FROM POOL  READ REQUSTS  WRITES
ISSUD      NEW BLKS
DLZSTTL ----I----------I----------I---------I-------------I------------I------------I-------------
I--------------
DLZSTTL    1          1          8       512              -             -             -
-             -
DLZSTTL    2          1         10      2048           1064           738           303
11            -
DLZSTTL    3          2         20      2048             39            32             6
-             -
DLZSTTL    4          2         32      4096              -             -             -
-             -
DLZSTTL
DLZSTTL ================================================================
DLZSTTL
DLZSTTL - INFORMATION PER DATA BASE -
DLZSTTL
DLZSTTL ----------(DL/I)---------I I------(VSAM)------I-----------(VSAM FOR HISAM,SHISAM AND INDEX
ONLY)------------
DLZSTTL
DLZSTTL DBD-    DMB-   SUB-  BUF- W   CI-    NUMBER OF   NUMBER OF   NUMBER OF    NUMBER OF   NUMBER OF
NUMBER OF
DLZSTTL NAME    ORGAN. POOL  SIZE    SIZE        EXCPS   CI-SPLITS   CA-SPLITS    RETRIEVES
UPDATES     INSERTS
DLZSTTL -------I------I----I-----I-I-----I------------I-----------I-----------I------------I-----------
I------------
DLZSTTL D08416A HDAM      1   512     512            -
DLZSTTL DD05264 HDAM      3  2048         (ACB CLOSED)
DLZSTTL KAHIDI  INDEX     4  4096         (ACB CLOSED)
DLZSTTL KAHIDAM HIDAM     4  4096         (ACB CLOSED)
DLZSTTL STDIX1P INDEX               2048            -           -           -            -
-             -
DLZSTTL STDIDBP HDAM      3  2048    2048            6
DLZSTTL STDCDBP HDAM      2  2048    2048          314
DLZSTTL STDCX2P INDEX               2048            2           -           -            1
-             1
DLZSTTL STDCX1P INDEX               2048           10           -           -           86
6             3
DLZSTTL -------I------I----I-----I-I-----I------------I-----------I-----------I------------I-----------
I------------
DLZSTTL SUBPOOL           2                        314
DLZSTTL SUBPOOL           3                          6
DLZSTTL -------I------I----I-----I-I-----I------------I-----------I-----------I------------I-----------
I------------
DLZSTTL TOTAL                                      332           -           -           87
6             4
DLZSTTL -------I------I----I-----I-I-----I------------I-----------I-----------I------------I-----------
I------------
DLZSTTL
DLZSTTL   N O T E
DLZSTTL      CI-SIZE         = CI-SIZE OF THE DATA BASE
DLZSTTL      BUF-SIZE        = BUFFER-SIZE OF THE SUBPOOL
DLZSTTL      W (WARNING)     = *, IN CASE CI-SIZE NOT EQUAL BUFFER-SIZE
DLZSTTL
DLZSTTL   R E M A R K S
DLZSTTL      THE VSAM STATISTICS VALUES ARE FROM THE LAST START OF THE DATA BASE.
DLZSTTL
DLZSTTL ================================================================
DLZSTTL     DL/I STATISTICS REPORT COMPLETE
DLZSTTL ================================================================
```

Figure 12. ...continued Example: Output of the DL/I Run Statistics Program (DLZSTTL)

If in this statistics an index component is assigned to a subpool, it means that this index database has an insert processing option. It does not mean that the index buffers are allocated to that subpool. Allocation of index buffers should be specified through:

```
   TYPE=BUFFER,HSBFR=...
```

Features Introduced with DL/I VSE 1.11

The following enhancements are available with DL/I VSE 1.11.

Multiple MPS Systems

The DL/I MPS Master Partition Controller DLZMPC00 can be activated in more than one CICS/DLI partition at the same time. An MPS batch job can select any active CICS/DLI online system and process databases defined there (one connection per job step). The selection is done by using new parameters of the DLI/DLR statement submitted with the MPS batch job.

Each MPS Master Partition Controller DLZMPC00 is started or stopped in its own partition and in the same way as before using the DL/I MPS Start/Stop transactions CSDA/CSDD.

New parameters for DLI/DLR statement

The new parameters of the DLI/DLR statement are shown below. Either one or both of them, separated by a comma, can be entered:

```
         PARTID=xx
```

denotes the CICS/DLI MPS partition, where the database(s) to be processed are defined.

```
         APPLID=yyyyyyyy
```

denotes the CICS (generic) applid of the CICS/DLI MPS partition, where the database(s) to be processed are defined.

```
   DLI,progname,psbname,PARTID=F2
   DLI,progname,psbname,APPLID=DBDCCICS,RC=YES
   DLR,progname,psbname,PARTID=F4,APPLID=PRODCICS
```

Figure 13. Examples of the PARTID and APPLID Parameters

The RC parameter can be placed anywhere after the psbname, together with or without the new PARTID/APPLID parameters.

If only one CICS/DLI MPS system is up, the new parameters are not required and the MPS batch jobs run as before.

Invocation through DLZCTRL

It is also possible to use the DLZCTRL program to dynamically test, if a batch job should run in the normal DL/I batch environment (DLZRRC00) or under the control of MPS (DLZMPI00). DLZCTRL has been introduced in *DL/I Resource Definition and Utilities* (SH24-5021-02).

In order to let DLZCTRL make the correct selection, add a PARM value to the EXEC DLZCTRL statement, as shown in this example:

```
// EXEC DLZCTRL,SIZE=1024K,PARM='PARTID=F4,APPLID=        '
```

The following syntax rules apply to the PARM value:

- It must have a fixed length of 25 characters.
- Columns 1-7 must contain the string PARTID=.
- Columns 11-17 must contain the string APPLID=.
- At least one parameter should be specified.

DLZCTRL will call either DLZMPI00 or DLZRRC00 according to the following rules:

1. **PARM value present with correct syntax:** DLZCTRL will search for the CICS/DLI MPS system specified in the PARM value. If the requested CICS/DLI MPS system is active, DLZCTRL will call DLZMPI00, else it will call DLZRRC00.

   When specifying a PARM value, it is important that the DLI/DLR statement for DLZMPI00 contains matching PARTID/APPLID parameters. The PARM value is only used to test if a requested CICS/DLI MPS system is up (pre-check). The actual connection to send data between the MPS batch partition (DLZMPI00) and the CICS/DLI MPS system (DLZMPC00/DLZBPC00) is established from the PARTID/APPLID parameters in the DLI/DLR statement.
2. **PARM value present with incorrect syntax:** DLZCTRL will issue an error message and terminate.
3. **PARM value not present:** DLZCTRL will test, if any CICS/DLI MPS system is active. If yes, DLZCTRL will call DLZMPI00 else it will call DLZRRC00.

   This does not ensure that the CICS/DLI MPS system requested through a succeeding DLI/DLR statement is active. An unsuccessful MPS batch execution might follow (DLZ089I) instead of a successful regular batch execution. Therefore a PARM value should always be added.

   Note: The batch initialization program DLZRRC00 will ignore any PARTID/APPLID parameters.

More than 4 GB for HD Databases

DL/I HD (HDAM/HIDAM) databases can be allocated on more than one VSAM ESDS data set. Up to five data sets can be defined. This increases the maximum database size to 20 GB (5 times 4 GB). The enhancement is transparent to user applications. The split into multiple data sets will be realized through an updated DBD definition and a reorganization or new initial load of the database:

Definition:

Each VSAM data set is represented through a DATASET macro statement in the DBD-generation job. The database segments are allocated on the VSAM data sets according to the following rules:

- The first DATASET statement is entered with all parameters as before. It determines, on which VSAM data set the root segment resides and all subsequently defined segments up to the second DATASET statement.
- The second DATASET statement denotes the VSAM dataset, on which all subsequently defined segments are stored. Any following DATASET statements and segment definitions are processed accordingly. Currently up to five DATASET statements are allowed.
- The second to the fifth DATASET macro statement have the following syntax:

```
         label     DATASET     DD1=filename
```

label
Can be defined on column 1, it has no effect.

filename
Denotes the 1-7 character VSAM symbolic file name.

Other parameters, which can be entered on the first DATASET statement are not supported. The DEVICE, BLOCK, SCAN and FRSPC values are taken from the first DATASET statement. This implies that all VSAM data sets must reside on the same DASD-type and have the same blocksize. Data set definition parameters (DEFINE CLUSTER job) of the VSAM data sets should correspond.

Figure 14 on page 50 shows an example of a DBD definition, which allocates the database STDCDBP on four VSAM data sets: The root segment STSCCST and all following segments defined up to segment STSCLOC are allocated on the primary data set STDCDBC. Segment STPCORD and any segments defined after it are allocated on the second data set ST2CDBC. Starting with segment STCCITM, all segments are allocated on the third data set ST3CDBC. The segments STSCSTA and STSCHIS will reside on the fourth data set ST4CDBC.

There are no restrictions regarding the definition of logical relationships or secondary indexing.

Database Load: The database can be split into multiple VSAM data sets in the following way:

1. **Initial Load through user application:**

   The DBD generation job of the affected database has to be implemented according to the new definition rules (DBDGEN with multiple DATASET statements). PCB definitions (PSBGEN) have not changed. The new DBD and any new or existing PSB execution blocks have to be re-created by running the DL/I Block Builder utility DLZUACB0. PSB execution blocks have to be re-built for all PSBs, which allow access to the affected database. After that a user application can load the database using the new control blocks.

   Note: The facility to load a database stepwise via separate load runs (Partial Database Load) is not supported for databases residing on multiple data sets.

2. **Reorganization:**

   The database has to be unloaded using the DL/I Unload Utility DLZURGU0 and the DBD definition in the existing format (one DATASET statement). After that, new DBD and PSB execution blocks have to be created as described above. Then the database can be reloaded into the new format using the DL/I Reload Utility DLZURGL0 and the new control blocks built.

   The reorganization utilities for the creation of logical relationships and secondary indexes DLZURPR0, DLZURG10 and DLZURGP0 run as before.

   Note: The Partial Database Reorganization utility (DLZPRCT1/2) is not supported for databases residing on multiple data sets.

Execution:

The allocation of a database on multiple data sets is transparent to applications. User programs do not have to be changed. All DL/I calls can be issued as before.

Additional DLBL statements are needed for all secondary data sets in DL/I batch and CICS/DLI startup jobs.

Utilities:

The DL/I utilities as documented in the DL/I publication Resource Definition and Utilities can be used in the same way, when databases are allocated on multiple data sets. There are no new or changed control statements. However, these changes in existing job streams are necessary:

- Additional DLBL statements must be supplied for all secondary data sets, whenever a utility directly accesses a database.
- When using the DL/I Image Copy utility DLZUDMP0, a 'D' control statement must be entered for each data set (cols 13-19), which has been defined for the database. The control statements can be submitted with one job step.
- When using the DL/I Recovery utility DLZURDB0, each data set of the database must be recovered in a separate job step. The data set names are entered on the 'S' control statement (cols 13-19) as before.

```
        DBD                                                     X
              NAME=STDCDBP,           DATA BASE DESCRIPTION NAME      X
              ACCESS=HDAM,            HIERARCHICAL DIRECT             X
              RMNAME=(DLZHDC10,       RANDOMIZING ROUTINE PHASENAME   X
              3,                      ROOT ANCHOR POINTS PER BLOCK    X
              100,                    ROOT ADDR. AREA HI RELATIVE BLK X
              600)                    INSERT BYTES LIMIT FOR RAA
      DATASET                                                   X
              DD1=STDCDBC,            DLBL FILE NAME - FIRST DATASET  X
              DEVICE=CKD,             DISK DEVICE                     X
              BLOCK=(4096),           VSAM CONTROL INTERVAL SIZE      X
              SCAN=2                  # CYLINDERS SCAN FOR ISRT SPACE
        SEGM                                                    X
              NAME=STSCCST,           SEGMENT NAME FOR CUST NAME/ADDR X
              PARENT=0,               IT IS A ROOT SEGMENT            X
              BYTES=106,              DATA LENGTH                     X
              POINTER=TWIN            PHYSICAL TWIN FWD ONLY
        FIELD                                                   X
              NAME=(STQCCNO,SEQ,U),   UNIQUE KEY FIELD  (CUST #)      X
              BYTES=6,                FIELD LENGTH                    X
              START=1,                WHERE IT STARTS IN SEGMENT      X
              TYPE=C                  ALPHAMERIC DATA
              .
              .
              .
        SEGM                                                    X
              NAME=STSCLOC,           SEGMENT NAME CUSTOMER LOCATION  X
              PARENT=STSCCST,         PARENT IS CUST. NAME/ADDR SEGM  X
              BYTES=106,              FIELD LENGTH                    X
              POINTER=TWINBWD         BOTH PHYS. TWIN FWD AND BWD
        FIELD                                                   X
              NAME=(STQCLNO,SEQ,U),   UNIQUE KEY FIELD (LOCATION #)   X
              BYTES=6,                FIELD LENGTH                    X
              START=1,                WHERE IT STARTS IN SEGMENT      X
              TYPE=C                  ALPHAMERIC DATA
              .
              .
              .
```

Figure 14. DBD Definition with Multiple DATASET Statements

```
DS2      DATASET                                                    X
               DD1=ST2CDBC             DLBL FILE NAME - SECOND DATASET
         SEGM                                                       X
               NAME=STPCORD,           SEGMENT NAME CUSTOMER ORDER  X
               PARENT=STSCLOC,         PARENT IS CUST. LOCATION SEGM X
               BYTES=55,               DATA LENGTH                  X
               POINTER=TWINBWD,        BOTH PHYS.TWIN FWD AND BWD   X
               RULES=(PPV)             LOGICAL RELATIONSHIP RULES
         FIELD                                                      X
               NAME=(STQCODN,SEQ,U),   UNIQUE KEY FIELD (DATE & ORD #) X
               BYTES=12,               FIELD LENGTH                 X
               START=1,                WHERE IT STARTS IN SEGMENT   X
               TYPE=C                  ALPHAMERIC DATA
               .
               .
               .
DS3      DATASET                                                    X
               DD1=ST3CDBC             DLBL FILE NAME - THIRD DATASET
         SEGM                                                       X
               NAME=STCCITM,           SEGMENT NAME LINE ITEM       X
               PARENT=((STPCORD,       PHYSICAL PARENT IS CUSTOMER ORD X
               SNGL),                  PHYS. CHILD FIRST PTR. ONLY  X
               (STPIITM,               LOGICAL PARENT IS ITEM INFORMAT X
               V,                      REQUIRED FOR IMS/VS UPWARD COMP X
               STDIDBP)),              LOG.PARENT IS IN ITEM DATA BASE X
               BYTES=38,               DATA LENGTH                  X
               POINTER=(TWINBWD,       BOTH PHYS. TWIN FWD AND BWD  X
               LTWINBWD),              BOTH LOG. TWIN FWD AND BWD   X
               RULES=(PVV)             LOGICAL RELATIONSHIP RULES
         FIELD NAME=STKCIIN,           INVENTORY ITEM NUMBER        X
               BYTES=6,                FIELD LENGTH                 X
               TYPE=C                  ALPHAMERIC DATA
*
DS4      DATASET                                                    X
               DD1=ST4CDBC             DLBL FILE NAME - FOURTH DATASET
         SEGM                                                       X
               NAME=STSCSTA,           SEGMENT NAME CREDIT STATUS   X
               PARENT=STSCCST,         PARENT IS CUST. NAME/ADDR SEGM X
               BYTES=24,               DATA LENGTH                  X
               POINTER=TWIN,           PHYSICAL TWIN FWD ONLY       X
               RULES=(,FIRST)          INSERT NEW SEGMS FIRST NOT LAST
*                                      NOTE THERE IS NO KEY FIELD
*                                      DESCRIBED FOR THE ABOVE SEGM
         SEGM                                                       X
               NAME=STSCHIS,           SEGMENT NAME CUSTOMER HISTORY X
               PARENT=STSCCST,         PARENT IS CUST. NAME/ADDR SEGM X
               BYTES=(130,53),         SEGMENT IS VARIABLE LENGTH   X
               COMPRTN=DLZSAMCP,       NAME OF COMPRESSION ROUTINE  X
               POINTER=TWINBWD         BOTH PHYS. TWIN FWD AND BWD
         FIELD                                                      X
               NAME=(STQCHDN,SEQ,U),   UNIQUE KEY FIELD (DATE & ORD #) X
               BYTES=12,               FIELD LENGTH                 X
               START=3,                WHERE IT STARTS IN SEGMENT   X
               TYPE=C                  ALPHAMERIC DATA
         DBDGEN                        REQUIRED TO MARK DBD END
         FINISH                        FOR SOURCE COMPAT WITH IMS/VS
```

Figure 15. ....continued DBD Definition with Multiple DATASET Statements

Support of CICS Transaction Server Storage Protection

When operating the CICS Transaction Server-DL/I online system with storage protection (STGPROT=YES in SIT), DL/I allocates its system resources in storage protected with a partition key. For example:

- Online nucleus DLZNUCxx (loaded by CICS Transaction Server)
- DBD and PSB control blocks
- Buffer pool
- Access modules

This prevents them from being inadvertently overwritten through online programs, provided the programs are running in user key. To run online programs with user key, you have to define in the CDS file:

- Transactions with **TASKDATAKEY=USER**
- Programs with **EXECKEY=USER**

More information on the CICS Transaction Server storage protection can be found in the related CICS documentation.

Conditional SVA Loading

DL/I DOS/VS 1.10 and DL/I VSE 1.11 use different levels of phases, which are not compatible. Therefore the use of DL/I phases from the SVA may be critical when running in a CICS coexistence environment.

For DL/I VSE 1.11 a new parameter is available in the DLZACT TYPE=CONFIG macro. It allows control of the loading of SVA-eligible DL/I phases during CICS/DLI online initialization:

SVA=NO|YES}
The default is NO.

When **SVA=NO** is specified, all DL/I phases will be loaded into partition space and used from there, even if they originally reside in the SVA.

When **SVA=YES** is specified, DL/I will use the phases from the SVA, if they reside there. Otherwise it will load them into partition space and use from there.

For further details, refer to “CICS Coexistence Environment” on page 58.

DL/I DOS/VS 1.10 Enhancements

The following enhancements are available with DL/I VSE 1.11. They have also been made available with APAR fixes for DL/I DOS/VS 1.10 previous to the shipment of DL/I VSE 1.11. The corresponding APAR numbers are shown in parenthesis.

Processing Options N and T for DL/I Batch Environment (PN30337)

The combination of processing option N or T with the processing option GO or GOP will also be effective in the DL/I batch environment.

When DL/I detects an invalid pointer during the access to a database segment (because of concurrent updates from a CICS/DLI online partition, for example), PROCOPT=N returns status code GG, and PROCOPT=T performs an automatic retry instead of issuing an error message and abnormally terminating the application.

A detailed description of the processing options N and T can be found in the DL/I publication Resource Definition and Utilities.

Change Accumulation with Selected Logfiles (PN43036)

The existing Change Accumulation process required that for every database for which DL/I log change records were passed to the Change Accumulation process, there had to be a corresponding DL/I database OPEN record present on the input logfile(s). If this was not the case, Change Accumulation assumed that some logfile might have been missing, issued message DLZ375I, and ended abnormally.

While this ensured that the set of input logfile(s) was complete, it also prevented the Change Accumulation from running with only a selected single logfile. For example, it was not possible to run Change Accumulation with the second (or higher) part of a large logfile only. This made prompt recovery actions difficult in some cases.

The Change Accumulation utility will now also accept logfiles that do not contain database OPEN records for all databases with log change records. When such a condition is detected the new message DLZ431I is issued instead of DLZ375I. This is a warning and should remind the user that some logfile(s) may be missing or an incorrect logfile has been mounted in the input stream.

DUMP Macro with RC=12 (PN47959)

In case of an abnormal termination, the DL/I modules DLZOLI00, DLZURDB0, DLZUDMP0, DLZTPRT0, DLZLOGP0, DLZURGL0 and DLZURRL0 will now call the VSE DUMP macro with RC=12 instead of RC=0.

This allows the usage of the ON $RC JCL statement to prevent the execution of subsequent job steps, if one of these modules has failed.

No Unload of Tape Work Files - DLZURG10/DLZURGP0 (PN56301)

When running the DL/I Prefix Resolution utility DLZURG10 and the DL/I Prefix Update utility DLZURGP0 a new control statement may be used to invoke different rewind options, when some of the utility workfiles have been assigned to tape, instead of unconditionally unloading the tape. This allows the submission of the complete reorganization utility job sequence (Pre-reorg, Unload, Reload, Prefix-resolution and Prefix-update) repetitively without operator intervention.

The usage of the new control statement is optional. When specified, it is active for the following work files, when they are assigned to tape:

```
-  DLZURG10:  WRKINxx  SYS013
              INTRMED  SYS010
-  DLZURGP0:  WORKFIL  SYS011
              INDXWRK  SYS014
```

The format is as follows:

```
  REW=option
```

Columns 1-4 must contain REW= and column 5 must contain one of these letters to indicate the wanted tape option:

```
    N - means no rewind
    R - means rewind only, and
    U - means rewind and unload.
```

No other columns are used. When the new control statement is not entered, tapes will be unloaded by default during termination of the utility as before.

Support for DL/I 31-Bit Applications (PN67649)

DL/I applications can run in 31-bit mode and above the 16 MB line without restriction. Compilers, which can be used to create DL/I 31-bit applications are:

- PL/I for VSE/ESA
- COBOL for VSE/ESA
- COBOL II
- High Level Assembler

31-bit execution is possible both for DL/I online and batch/MPS applications, implemented with the DL/I CALL or HLPI interface. DL/I call parameters can reside above the 16 MB line.

User-written DL/I exit routines (such as randomizing and compression routines) cannot run in 31-bit mode.

Appendix A, “Sample Job Streams for DL/I 31-Bit Applications,” on page 107 provides sample job streams which show how to create DL/I 31-bit applications in different environments.

Additional Support for LE/VSE

The following LE/VSE functions interacting with DL/I are supported:

- TRAP runtime option
- CEETDLI interface

LE/VSE TRAP Runtime Option

DL/I batch applications implemented in one of the LE/VSE-conforming languages can run with the LE/VSE TRAP runtime option being active. When TRAP(ON) is set, any abnormal termination condition caused by the application will first be processed by LE/VSE error routines. If DL/I is to receive control again after LE/VSE error handling, for example, to write back pending DL/I data buffers and correctly close the DL/I databases or log files:

```
ABTERMENC(ABEND)
```

or any equivalent runtime option must have been specified. Only this ensures proper DL/I termination, when TRAP(ON) has been set.

When running with TRAP(ON), DL/I STXIT linkage is overlaid by LE/VSE STXITs. Any DL/I dumps issued by one of the DL/I exits after message DLZ001I will not reflect the original storage contents from the time, when the error had occurred. These DL/I dumps can be suppressed by the according UPSI byte setting as described in Resource Definition and Utilities.

When original DL/I STXIT linkage and exit processing is preferred, the application should be run with TRAP(OFF). Information on DL/I STXIT linkage can be found in the DL/I publications Resource Definition and Utilities and Guide for New Users.

For information on the LE/VSE TRAP option, refer to the corresponding LE/VSE documentation.

LE/VSE CEETDLI Interface

LE/VSE provides a callable service, CEETDLI, that can be used to issue DL/I calls. CEETDLI in general performs the same functions as the existing language-specific interfaces (ASMTDLI, CBLTDLI, PLITDLI), but is language independent. Only LE/VSE conforming applications can call CEETDLI. Calls to CEETDLI are implemented in the same way as calls to the language-specific interfaces. A description of the CEETDLI syntax can be found in the LE/VSE Programming Guide.

The existing language-specific interfaces will continue to be supported both in existing programs and programs newly compiled and linked in the new LE/VSE environment.

Increased VSAM Blocksize for DL/I Index Databases (PN68583)

The maximum blocksize for DL/I index databases (VSAM KSDS) has been increased to 30720 bytes. During DBD-generation DL/I calculates the blocksize from the values entered at the BLOCK and RECORD parameters of the DATASET statement. New possible blocksizes are:

- 512 bytes - 8192 bytes in steps of 512 bytes
- 8192 bytes - 30720 bytes in steps of 2048 bytes

The maximum segment size has not been changed.

Command Code U for HLPI Programs (PN73378)

The command code U will also be supported in the DL/I HLPI language through the new option KEEP.

KEEP can be used with all types of DL/I GET calls and the DL/I ISRT call. It requires the same position as the existing options FIRST or LAST. It should not be specified on the same segment level together with FIRST or LAST. Coding example:

```
EXEC DLI GET NEXT IN PARENT USING PCB(2)
                FIRST SEGMENT (SEGMX)
                KEEP  SEGMENT (SEGMY)
                      SEGMENT (SEGMZ) INTO (IOAREA)
```

A general description of the command code U can be found on page 4-3 of the DL/I CALL and RQDLI Interfaces publication (SH12-5411-6). The same rules and restrictions mentioned there also apply to the HLPI-style usage. The DL/I HLPI language is documented in the DL/I HLPI publication (SH24-5009-2).

New System Call QURY (PN85936)

The new DL/I system call QURY is available for DL/I online applications. It returns the status of a DL/I database.

The DL/I sample program for system calls, DLZMDLI0, allows the selection of the new QURY call.

Year 2000 Support (PN87288)

DL/I has to process date values when they are passed as part of:

- log records
- utility records (like file header or detail record)
- utility control statements.

To support Year 2000, the existing 2-digit year values of the mentioned DL/I records and control statements will be interpreted according to a fixed-window scheme:

```
yy = 50->99 = 19yy
yy = 00->49 = 20yy
```

The date format of the physical records and control statements remains unchanged. All date values printed by DL/I utilities (such as page headers, log print records, certain start messages) will show the year in a 4-digit format.

Access to a DL/I database is not affected by the Year 2000 conversion.

GETVIS for Program Isolation Storage Above 16 MB (PN88972)

Depending on the DL/I application design (e.g type of database calls used and TERM/CHKP call frequence) the program isolation function may use up to several megabytes of working storage. DL/I will now always attempt to obtain this working storage from the GETVIS-31 area. When not enough GETVIS-31 storage is available, it will be requested from the GETVIS-24 area as before.

More Than 32 HD-Buffers per Subpool (PN89468)

The maximum number of HD-buffers for VSAM ESDS data sets, which can be allocated in a subpool has been increased to 255. The definition of the wanted number of HD-buffers is done as before:

- In the DL/I batch environment via the HDBFR statement in the DLI parameter statement.

```
HDBFR=(nn,dbdname1,dbdname2,..)
```

- In the DL/I online environment via the DLZACT TYPE=BUFFER macro in the ACT generation job.

```
DLZACT TYPE=BUFFER,HDBFR=(nn,dbdname1,dbdname2,...)
```

The number of buffers nn can be specified in a range from 2 to 255. The default value of 32 is unchanged.

PSBs Above 16 MB (PQ09904)

During CICS/DLI initialization, PSBs may be loaded into GETVIS-31 storage of a CICS/DLI online partition. The location of the PSBs is controlled through the new PSBLOC parameter of the DLZACT TYPE=CONFIG macro in the ACT generation job:

```
PSBLOC={ANY|BELOW}
```

When ANY is specified, DL/I will try to load the PSBs into GETVIS-31 storage. If there is not enough GETVIS-31 storage available, the PSBs will be loaded into the GETVIS-24 area.

When BELOW is specified, the PSBs will be loaded into the CICS DSA as before. This is the default.

When the PSBs are residing above the 16 MB line, every CICS/DLI task scheduling a PSB will get its own copy in the DSA. That means the original PSB is copied over from GETVIS-31 to the DSA and processed there. This is necessary, because the DL/I action modules are executing in AMODE-24. At DL/I task termination the storage of the PSB copy is returned to the DSA.

In this way, the whole set of PSBs defined in the DL/I online nucleus and initially loaded during CICS/DLI startup, can be held in GETVIS-31 storage. DSA storage for PSBs will only be needed on demand from a DL/I task.

Migrating to DL/I VSE 1.11 and the CICS Transaction Server for VSE/ESA 1.1

DL/I VSE 1.11 is upward compatible from DL/I DOS/VS 1.10. Programming interfaces have not been changed. DL/I batch applications and online applications implemented in CICS command level language will run as before. CICS online applications implemented in CICS macro style or accessing CICS control blocks directly need to be converted as shown in “CICS Transaction Server for VSE/ESA 1.1 Migration Items” on page 57.

Migration aspects related to the enhancements of DL/I VSE 1.11 are described below. If you are migrating from an earlier DL/I release than DL/I DOS/VS 1.10, you should also refer to Chapter 4, “Migration and Compatibility Considerations,” on page 15.

For specific migration aspects concerning the CICS Transaction Server, refer to “CICS Transaction Server for VSE/ESA 1.1 Migration Items” on page 57.

DL/I VSE 1.11 Migration Items

Changed Modules of DL/I Online Nucleus

All DL/I online users must reassemble the online nucleus with the DL/I DLZACT macro.

Storage Layout Control Table

The use of the DL/I Storage Layout Control (SLC) table is no longer supported.

Processing Option N for DL/I Batch Environment

PROCOPT=N returns status code GG instead of abnormal termination if invalid DL/I pointers are detected. DL/I batch applications might must be changed to examine the new status code.

Change Accumulation with Selected Logfiles

The completeness check for input log files of the Change Accumulation Utility has been changed. Refer to “Change Accumulation with Selected Logfiles (PN43036)” on page 52 for details.

Dump Macro with RC=12

Some DL/I programs as described in “DUMP Macro with RC=12 (PN47959)” on page 52 are now terminating with RC=12 instead of RC=0, when an abend situation has been detected. Jobs using conditional JCL may have to be adapted accordingly.

Support for DL/I 31-Bit Applications

DL/I applications, which are to exploit the 31-bit support of the new LE/VSE compilers or the High Level Assembler for VSE, must be recompiled and relinked as described in Appendix A, “Sample Job Streams for DL/I 31-Bit Applications,” on page 107.

Existing applications (24 or 31 bit addressing) run as before.

You must specify PATH sensitivity for all segments, for which you do path retrieve, insert or replace calls from 31-bit applications. This applies both for CALLDLI and EXEC DLI statements. A new PSBGEN and ACBGEN must be run for all affected PSBs. For changes required for the new CICS Transaction Server online environment, refer to “CICS Transaction Server for VSE/ESA 1.1 Migration Items” on page 57.

LE/VSE TRAP Runtime Option

The abend handling for DL/I batch and MPS batch programs works slightly differently, when running with the LE/VSE runtime option TRAP(ON). Refer to the description of the LE/VSE Trap Runtime Option.

Increased VSAM Blocksize for DL/I Index Databases

When the DATASET statement of an index DBD has been changed to define a different blocksize, a new DBDGEN must be done for this index database. Then ACBGEN must be rerun for any PSBs, which may access the database.

PSBs above 16 MB

The new PSBLOC parameter becomes active by reassembling the online nucleus with the DL/I DLZACT macro.

Change of Support for DL/I IMF and IUG

The Interactive System Productivity Facility (ISPF), a prerequisite for using the DL/I IMF and IUG functions, is no longer supported since VSE/ESA.

Users operating VSE under the control of VM can continue to use DL/I IMF and IUG through ISPF running on CMS.

CICS Transaction Server for VSE/ESA 1.1 Migration Items

Applications

CICS/DLI macro level programs must be converted to the CICS command level.

Adaptions are also required for CICS/DLI online programs which have been written using the DL/I CALLDLI interface, and which are using the CICS TCA to check return codes (TCAFCTR) or retrieve the PCB address list (TCADLPCB). This is needed because the CICS TCA can no longer be accessed. The equivalent fields can be retrieved from the DL/I UIB control block instead.

The publication DL/I CALL and RQDLI (SH12-5411-6) contains examples showing the syntax for DL/I calls, when the UIB is used.

CICS Transaction Server - DL/I Tables Requirements

The CICS components needed for DL/I online execution must be defined according to the description in DL/I Resource Definition and Utilities (SH24-5021-02).

The CICS PCT and PPT tables no longer exist. These entries must be specified using the CICS RDO facility.

All DL/I user and system programs should be defined with EXECKEY(USER) and the associated transactions with TASKDATAKEY(USER). The MPS master partition controller DLZMPC00 no longer needs a definition for a CICS transaction workarea (TWA) in the CSDB transaction entry.

Specify DBP=2$ in the CICS SIT to allow for dynamic transaction backout for DL/I programs.

CICS Coexistence Environment

Starting with VSE/ESA 2.4, the CICS Transaction Server 1.1 and CICS/VSE 2.3 can coexist and run together with DL/I VSE 1.11 (CICS Transaction Server) and DL/I DOS/VS 1.10 (CICS/VSE).

In such an environment you can make use of the following functions in addition to regular CICS/DLI online transaction processing:

- CICS/DLI MRO function shipping for usage of remote or extended remote PSBs.
- Execution of MPS batch jobs. The release level of the DLZMPI00 root phase must correspond to the release level of the MPS master partition controller DLZMPC00 and the CICS/DLI online system it tries to connect to. MPS batch jobs from different releases may run at the same time.

DL/I DOS/VS 1.10 and DL/I VSE 1.11 are installed in separate sublibraries. Therefore, the correct LIBDEF chain is essential for loading the correct version of the DL/I modules.

In a coexistence environment, you should load SVA-eligible phases from the DL/I DOS/VS 1.10 sublibrary, ensuring that the DLZACT for DL/I VSE 1.11 has the parameter **SVA=NO** set. For further information about the SVA parameter, refer to “Conditional SVA Loading” on page 52.

Features Introduced with DL/I VSE 1.12.0

You can use DL/I VSE 1.12.0 to reduce storage contention below the 16 MB line. Those of its resources that had usually held a larger amount of storage in the 24-bit area can now be loaded into 31-bit GETVIS space above the 16 MB line.

Allocation of DL/I Resources above 16 MB

The following DL/I resources can be allocated above the 16 MB line of storage when running in a CICS/DLI online environment:

- PSBs
- HD data buffers
- Most of the DL/I action modules and the MPS mirror program DLZBPC00
- DL/I user exit routines for segment compression, field sensitivity and secondary indexing

As described in “DL/I DOS/VS 1.10 Enhancements” on page 52, allocating the PSBs above the 16 MB line during initial load in a CICS/DLI online environment has already been possible with DL/I DOS/VS 1.10 and DL/I VSE 1.11 after installing APAR PQ09904. The PSB copies for the single transactions however were still created in 24-bit storage below the line. This has been extended, so that PSB copies can reside above 16 MB as well.

In the regular DL/I batch environment, PSB, HD buffers, DL/I action modules, and user exit routines are allocated in the 24-bit area as before.

When the DL/I action modules or user exit routines have been loaded into the SVA (low or high), they will be used from there, both in the DL/I batch and online environment.

Allocation of PSBs

You can define the residence of DL/I PSBs in a CICS/DLI online environment with the already existing PSBLOC parameter of the DLZACT TYPE=CONFIG online generation macro:

PSBLOC= [ANY] | BELOW | [GV24]

ANY
loads the PSBs into 31-bit GETVIS storage during CICS/DLI online initalization. This function is available with APAR PQ09904 since DL/I DOS/VS 1.10 and DL/I VSE 1.11. Refer to “DL/I VSE 1.11 Enhancements” on page 61 for details.

In addition, the working copies of the PSBs for the single transactions are also allocated in 31-bit GETVIS space

- for application programs scheduling the PSB in AMODE 31
- for application programs scheduling the PSB in AMODE 24, if they are implemented in the DL/I HLPI i/f, because HLPI programs do not access the PCB, which is the only part of the PSB that an application can reference.

When scheduling is done in AMODE 24 and the application program is implemented in the DL/I call i/f (which needs access to the PCB), the PSB copy will be allocated below the 16 MB line in the CICS CDSA. An exception is the MRO/ISC mirror program DFHMIRS, entering DL/I in AMODE 24 with the DL/I call i/f. This will still receive the PSB copy in 31-bit GETVIS space, because it is running in AMODE 31 (except for submitting the DL/I calls).

Note: Load PSBs and PSB copies are only allocated in the 31-bit area, if you have also specified the new parameter USREXIT=AMODE-31. This parameter is described "Addressing Mode of DL/I User Exit Routines" on page 59.

BELOW
CICS CDSA storage below 16 MB is used during PSB initial load and for the PSB working copies. This is the default.

GV24
PSBs are loaded into 31-bit GETVIS storage during CICS/DLI online initialization. The PSB working copies for the single transactions are allocated in 24-bit GETVIS space.

In the DL/I batch environment the PSBLOC parameter is not supported. The PSB is loaded into program storage below the 16 MB line as before.

Allocation of HD Buffers

You can define the residence of DL/I HD buffers in a CICS/DLI online environment with the new HDMODE parameter of the DLZACT TYPE=CONFIG online generation macro:

HDMODE= [ANY] | BELOW

ANY
allocates HD ESDS buffers in 31-bit GETVIS storage

Note: HD buffers are only allocated in the 31-bit area if you have also specified the new parameter USREXIT=AMODE-31. This parameter is described in "Addressing Mode of DL/I User Exit Routines" on page 59.

BELOW
CICS CDSA storage below 16 MB is used for HD buffer allocation. This is the default.

In the DL/I batch environment the HDMODE parameter is not supported. All HD buffers are allocated in program storage below the 16 MB line as before.

Addressing Mode of DL/I User Exit Routines

When PSBs or HD buffers reside above 16 MB in a CICS/DLI online environment, addresses that are passed to DL/I user exit routines (randomizing, segment compression, field sensitivity or secondary indexing), by way of control blocks or registers, can also point to data above 16 MB. In this situation DL/I calls the exit routines in AMODE 31, regardless of their library attributes, so that they are able to access these data. Specifically during the migration phase it is not sure, if existing exit routines with AMODE 24/RMODE 24 library attributes can also run successfully in AMODE 31. Therefore, you have to confirm the addressing mode by using a new parameter of the DLZACT TYPE=CONFIG online generation macro:

USREXIT= [AMODE-31] | AMODE-24

AMODE-31
indicates that DL/I user exit routines are capable to run in 31-bit addressing mode, regardless of their library attributes. Defining AMODE-31 for user exit routines is a requirement that PSBs or HD buffers can be allocated above the 16 MB line by using the parameters PSBLOC=ANY or HDMODE=ANY.

If you have specified AMODE-31 and PSBs or HD buffers are residing in 31-bit address space, DL/I will invoke the user exit routines in AMODE 31, even if they, for example, have the library attributes AMODE 24/RMODE 24.

When PSBs and HD buffers are residing in 24-bit address space, DL/I invokes the user exit routines in the addressing mode of their AMODE library attribute.

AMODE-24
must be specified if DBDs or PSBs include DL/I user exit routines that need to run in 24-bit addressing mode.

AMODE-24 resets PSBLOC=ANY to PSBLOC=BELOW and HDMODE=ANY to HDMODE=BELOW. This causes PSBs and HD buffers to be loaded below the 16 MB line. In addition, the parameters from DL/I 31-bit applications are also copied below 16 MB. After that, all DL/I data reside in 24-bit address space and DL/I invokes the user exit routines in the addressing mode of their AMODE library attribute. This is the default.

Note: Exit routines provided by the DL/I distribution libraries (DLZHDC10, DLZHDC20, DLZHDC30, DLZHDC40, DLZSAMCP) are delivered with the library attribute AMODE 31 and can run in 31-bit addressing mode. This allows to set USREXIT=AMODE-31.

The parameter USREXIT is not supported in the DL/I batch environment. All DL/I data reside below 16 MB (parameters from DL/I 31-bit applications are copied down), and the DL/I user exits are called in the addressing mode of their AMODE library attribute.

When PSBs or HD buffers are residing below the 16 MB line, DL/I randomizing routines are called in AMODE 24.

Allocation of DL/I Action Modules and User Exit Routines

The DL/I action modules (except the DL/I open/close routine DLZDLOC0), and the DL/I provided user exit routine for segment compression DLZSAMCP now have the library attributes AMODE 31/RMODE ANY. Except the randomizing routines, exit routines created by the user have no restriction anymore to 24-bit execution or residence, but they can have any valid AMODE/RMODE combination and can therefore also be defined for addressing mode 31 and residence above 16 MB. Randomizing routines are not restricted in respect of their addressing mode, but, for compatibility reasons, still need to have residence mode 24. The DL/I provided randomizing routines DLZHDC10, DLZHDC20, DLZHDC30, and DLZHDC40 are defined with AMODE 31/RMODE 24.

In a CICS/DLI online environment, the DL/I action modules (except the DL/I open/close routine DLZDLOC0) are loaded into 31-bit GETVIS space, or used from the SVA-high, if they reside there. DLZDLOC0 has AMODE 31/RMODE 24 and is allocated in CICS CDSA storage below the line. DL/I user exit routines are loaded according to their RMODE library attribute into 24-bit CDSA (RMODE 24) or 31-bit GETVIS (RMODE ANY) space. When residing in the SVA (low or high) they are used from there. DL/I randomizing routines are always loaded into the CICS CDSA below the line, even if they have RMODE 31 and can only be used from the SVA-low.

In the DL/I batch environment, the DL/I action modules and user exit routines are loaded into program storage below 16 MB as before, unless they reside in the SVA (low or high). In this case they are used from there. DL/I randomizing routines can only be used from the SVA-low.

DL/I GETVIS Storage in Separate Subpools

GETVIS storage allocation for different types of DL/I resources is assigned to different subpools. The following GETVIS subpool IDs are used:

DLIPSB
PSBs loaded during CICS/DLI initialization

DLIWPS
PSB working copies for the single transactions

DLIBFR
buffer pool for HD databases

DLIMOD
action modules and user exit routines

DLIQWA
queuing elements of the Program Isolation function

You can query the amount of storage that is held by the different subpools by using the GETVIS attention command.

Conditional SVA Loading Disabled

Until z/VSE 4.2 it was possible to run CICS/VSE 2.3 and DL/I DOS/VS 1.10 on the same VSE system, in a coexistence environment together with CICS Transaction Server for VSE/ESA and DL/I VSE 1.11.

As DL/I DOS/VS 1.10 and DL/I VSE 1.11 were using different levels of phases, which are not compatible, their use from the SVA was critical when both DL/I releases were running concurrently. For this reason, the SVA= parameter of the DLZACT TYPE=CONFIG macro was available for DL/I VSE 1.11, allowing to control the loading of SVA-eligible phases during CICS/DLI online initialization as described in “Conditional SVA Loading” on page 52.

This SVA= parameter is not supported anymore. Starting with z/VSE 4.3, the CICS coexistence environment is not supported anymore and DL/I VSE 1.12 is the only DL/I release that can be used.

DL/I VSE 1.11 Enhancements

The following enhancements are available with DL/I VSE 1.12.0. They have also been made available previously with APAR fixes for DL/I VSE 1.11. The corresponding APAR numbers are shown in parentheses.

Return UIB Address in User DIB at EXEC DLI (PQ37331)

In the HLPI (EXEC DLI) programming environment, DL/I status information is returned in the variables of the DL/I Interface Block (User DIB). This is different from the DL/I call-level programming environment, where DL/I status information is primarily returned through the PCB control block.

When a CICS/DLI online program implemented with the HLPI interface has scheduled a PSB and invokes another program implemented with the call-level interface that should use the same PSB and the environment is CICS/VSE 2.3 with DL/I DOS/VS 1.10, the call-level program could use the field TCADLPCB from the CICS TCA to retrieve the addresses of the PCBs.

As with the CICS Transaction Server access to the TCA control block is no more possible for application programs, the new field DIBUIBA, has been introduced in the user DIB. This allows the call-level program to obtain the PCB addresses in the following way:

1. The HLPI program schedules the PSB and retrieves the address of the UIB control block through field DIBUIBA.
2. The HLPI program invokes the call-level program passing the address of the UIB to it.
3. The call-level program obtains the addresses of the PCBs through field UIBPCBAL.

When the call-level program has retrieved the PCB addresses, it can enter DL/I calls to the same PCBs as the HLPI program, which had originally scheduled the PSB.

Allow variable for PSB name in DL/I HLPI Scheduling call (PQ39433)

The HLPI (EXEC DLI) syntax of the DL/I Scheduling call has been extended that in addition to constants, variables are also supported to specify the name of the PSB to be scheduled. When a variable is used, it must be enclosed in double parentheses as shown below:

```
COBOL:     EXEC DLI SCHEDULE PSB((psb_name)) END-EXEC.
PL/I:      EXEC DLI SCHEDULE PSB((psb_name));
```

psb_name is the variable containing the name of the PSB to be scheduled.

DL/I AIBTDLI Interface (PQ39683)

The AIBTDLI interface allows programs running in a VSE batch partition to issue DL/I calls without a DL/I batch environment having been established using DLZRRC00 or DLZMPI00.

A detailed description of this interface can be found in the z/VSE e-business Connectors, User's Guide.

DSA Fragmentation and SOS with PSBLOC=ANY (PQ51026)

Long running CICS/DLI online transactions issuing frequent CICS syncpoint calls can cause a CICS SOS condition. This can happen, when the CICS DSA, through repeated freemain/getmain sequences implied from the syncpoint calls, becomes fragmented in a way that gaps are not large enough to hold a new copy of the PSB to be re-scheduled.

The problem was initially reported for DL/I DOS/VS 1.10 and CICS/VSE 2.3 with APAR PQ29613. Although the storage manager of the CICS Transaction Server for VSE/ESA (the online platform for DL/I VSE 1.11) can work differently from the storage manager of CICS/'VSE , which is the required platform for DL/I DOS/VS 1.10, it can happen that this type of storage fragmentation can also occur with DL/I VSE 1.11 and the CICS Transaction Server. Therefore, the circumvention of the problem, the option PSBLOC=GV24 of the DLZACT TYPE=CONFIG macro (originally provided for DL/I DOS/VS 1.10), has also been made available for DL/I VSE 1.11.

If you have specified PSBLOC=GV24, PSBs will be loaded into 31-bit GETVIS storage during CICS/DLI initialization in the same way as with PSBLOC=ANY. The difference is that the copy of the PSB being scheduled will not be allocated in the CICS DSA, but in the 24-bit partition GETVIS area and therefore it is not affected if a fragmentation of the DSA should occur.

More than 25 Characters in PARM-String for DLZCTRL (PQ54003)

The 'PARM=' string passed to the DL/I dynamic initialization program DLZCTRL via the EXEC statement, can now also have more than 25 characters, if the DL/I specific part of it - the first 25 characters - corresponds to the syntax rules as documented in “Multiple MPS Systems” on page 47. DLZCTRL does not check the text following the 25th character. This part is now available to be interpreted by the program that is executed.

Two examples of valid 'PARM=' strings are shown in the following figure:

```
PARM='PARTID=F2,APPLID=DBDCCICS'                (old format with 25 characters)
PARM='PARTID=F2,APPLID=        /anytext...'     (new extended format)
```

Figure 16. Examples of Valid 'PARM=' strings

DTIMOUT does not work for DL/I BBCALL waits (PQ65658)

On the CICS Transaction Server platform, DL/I waits are normally issued via the CICS Black Box (BBCALL) interface. These waits are not sensitive to the CICS DTIMOUT parameter and therefore CICS/DLI tasks will not be purged at a timeout condition while they are suspended in a BBCALL wait.

When the system runs under heavy load with many concurrent CICS/DLI tasks, the DL/I Program Isolation (PI) function might have to serialize DL/I database accesses in a way that many tasks are held in a (BBCALL) wait for a longer period of time and the CICS/DLI system appears like being in a hang.

In order to allow that PI-waits can be resolved automatically by purging DL/I tasks through the CICS timeout feature after a certain wait time, a new DL/I parameter has been introduced for the DLZACT TYPE=CONFIG macro:

```
PTIMOUT={YES|NO}
```

When PTIMOUT=YES is specified, the CICS DTIMOUT parameter will also be effective for DL/I PI-waits. All other types of DL/I (BBCALL) waits are not affected and will not be DTIMOUT-purged as before.

When PTIMOUT=NO is specified, the CICS DTIMOUT parameter will not be effective for DL/I (BBCALL) waits as before.

The default is NO.

This APAR requires that the corresponding APAR PQ65543 of the CICS Transaction Server has also been installed.

DL/I Support for LE/VSE UADUMP Option (PQ68093)

When the LE/VSE UADUMP option is set, LE/VSE may create a system dump of the user partition when an abnormal termination has occurred. At the same time, if it was a DL/I batch application, DL/I dump processing is controlled through the UPSI statement, which may independently create a partition dump as well.

APAR PQ68093 prevents that both LE/VSE and DL/I take a partition dump while allowing full LE/VSE and DL/I abend handling to take place. When UADUMP has been defined and it is a DL/I batch application, the user should also request DL/I abend handling and a partition dump through UPSI bits 5 and 7 (set to zero). If an abnormal termination occurs with these settings, either LE/VSE writes the partition dump and the DL/I abnormal termination routine suppresses the duplicate one, or, if LE/VSE was not able to take the dump, DL/I will do it instead.

Misleading Result Information in DLZMDLI0 for QURY Call (PQ81067)

The result information returned from the DL/I online test program DLZMDLI0 for the QURY-function has been changed. The RESULTS-field on the screen will now display

STARTED
when the database was started, or

STOPPED
when the database was stopped

In both cases, a blank UIB status will be shown.

Scheduling Error when PSB is Generated with LANG=PLI (PQ92038)

The restriction that PSBs of language-type PL/I (LANG=PLI parameter in PSBGEN statement) are not supported for programs using the AIBTDLI interface has been removed. In addition, there is no dependency between the language of the PSB and the language of the program when the AIBTDLI interface is used. For programs implemented in the normal DL/I HLPI or call-level interface, the same rules apply as before.

For a description of the AIBTDLI interface, refer to the z/VSE e-business Connectors, User's Guide.

GETVIS-24 PSB Copies Allocated in Separate Subpool (PK15772)

When the parameter PSBLOC=GV24 has been defined in the DL/I online nucleus generation (DLZACT TYPE=CONFIG macro), the copies of the PSBs being scheduled are allocated in the 24-bit area of the partition GETVIS storage. In situations of heavy workload or perhaps specific unexpected conditions, the CICS/DLI system may more and more use up this 24-bit GETVIS storage or even run out of it. In this case,

it is helpful to determine how much of this space has been allocated by DL/I, so that appropriate actions can be taken.

For this reason, the DL/I GV24 PSB-copies will now be allocated in their own separate subpool with the name DLIPSB. This allows to tell, for example, via the attention GETVIS command, how much of the 24-bit GETVIS storage has been consumed by DL/I at a certain point of time.

Relational Operators in Alphabetical Notation (PK25497)

The relational operators of a DL/I SSA can now, in addition to their mathematical notation, also be specified in alphabetic notation.

Table 5. Relational Operators of a DL/I SSA

Mathematical Notation	Alphabetical Notation
' ='	'EQ'
'=>'	'GE'
'=<'	'LE'
' >'	'GT'
' <'	'LT'
' ¬='	'NE'

Note: The two characters can also be used in reverse order, for example, '= ', '>=', ...

Allow Duplicate Secondary Indexes during Prefix Update (PK44598)

To help reorganizing databases with duplicate secondary indexes, there is a new option to let the DL/I prefix update utility tolerate duplicate secondary index entries instead of cancelling the job. The reorganization process terminates successfully and duplicate secondary index conflicts can be analyzed later by using the reorganized version of the database. Whether duplicate secondary indexes should be accepted during the prefix update run can be selected with the new input control statement DPLINDEX.

DPLINDEX=YES | NO

Both with DPLINDEX=YES and DPLINDEX=NO all workfile records are read even when duplicate secondary index entries are encountered. When a duplicate secondary index situation occurs, the DL/I prefix update utility will always insert the first one of the duplicates into the index database and skip the others. Any workfile records which were skipped (that is, those records following the first one of the duplicates), are printed on SYSLST together with messages DLZ004I and DLZ960I, which are additionally written to SYSLOG.

If you have specified DPLINDEX=NO - and duplicate secondary indexes were found - the DL/I prefix update utility will, at the end of its execution, abend and result in an unsuccessful reorganization. This is the default and works as before.

If you have specified DPLINDEX=YES - and duplicate secondary indexes were found - the DL/I prefix update utility will terminate normally with RC=4, and result in a successful reorganization. The first workfile record of a duplicate secondary index situation (the one actually selected for secondary index insertion), is also printed and written in front of the other duplicates. In addition, an indication is shown with each duplicate record, if it was the one selected or one that had been skipped. With the information provided with the workfile records you can then determine, after the reorganization has been completed, if you want to keep those of the duplicates as selected by the prefix update utility – first one retrieved from the workfile as described above - or make another choice by manually updating the target pointers in the secondary index records.

If used, the DPLINDEX control statement must begin at column 1 and can be entered before or after the 'U' control statement.

Up to 10 Data sets per DL/I HD Database (PK48347)

Starting with DL/I VSE 1.11 a DL/I HD database can be allocated on up to 5 VSAM ESDS data sets as described in “More than 4 GB for HD Databases” on page 48. This has been extended to a maximum allocation on up to 10 VSAM ESDS data sets with a total capacity of 10 * 4 = 40 GB.

DL/I Reload with Empty Database (PK49053)

It is possible to unload a DL/I HD database that contains no segments, for example, when all segments have been deleted, creating an Unload file that consists of the statistics records only. When this Unload file was passed to a later Reload step, the DL/I Reload utility abnormally terminated with message DLZ387I and the database has not been rebuilt.

However, the database reorganization procedure of a DL/I user can depend on a successful execution of the DL/I Reload utility even when the Unload file contained no segments and require that a logically empty database (with a VSAM HURBA unequal 0) is then built, to which segments can be added later on.

Now, this can be achieved by specifying the following new control statement:

NOSEGMTS=BUILD

Depending on the database organization the empty database consists of

HDAM
The control record (block 0) only

HIDAM
The control record (block 0), the first bit map (block 1), and the x'FFFF..' (= EOD) segment in block 2

The empty database can then be taken into regular operation and, for example, be loaded through DL/I Insert calls. If you have specified NOSEGMTS=BUILD, message DLZ387I is issued as a warning instead of an error and the job ends successfully with RC=4 instead of unsuccessfully with RC=12. Finally message DLZ388I is issued.

If used, the NOSEGMTS=BUILD control statement must begin at column 1.

Migrating to DL/I VSE 1.12.0

DL/I VSE 1.12.0 is generally upward compatible from DL/I VSE 1.11. Programming interfaces have not been changed. DL/I batch and online applications should run as before.

Migration aspects related to the enhancements of DL/I VSE 1.12.0 are described in “Migration Items from New DL/I VSE 1.12.0 Functions ” on page 65. Considerations required due to new functions introduced with APAR fixes for DL/I VSE 1.11 previous to the shipment of DL/I VSE 1.12.0 are shown in “Migration Items from DL/I VSE 1.11 APARs ” on page 67.

If you are migrating from an earlier DL/I release than DL/I VSE 1.11, see Chapter 4, “Migration and Compatibility Considerations,” on page 15.

Migration Items from New DL/I VSE 1.12.0 Functions

The following section describes the migration items resulting from the new DL/I VSE 1.12.0 functions as described in “Features Introduced with DL/I VSE 1.12.0” on page 58.

Application Control Blocks

A new DBDGEN or PSBGEN assembly is not required. The executable format of the DBDs (DMBs) has not changed.

Execution control blocks for all PSBs must be recreated with the DL/I block builder utility DLZUACB0. Use parameter DMB=YES to invoke this utility.

PSB execution control blocks created before DL/I VSE 1.12.0 are not compatible and will be rejected by issuing message DLZ017I (batch) or message DLZ071I (online).

DL/I Online Nucleus

Online users must reassemble and recreate a new DL/I online nucleus by using the DLZACT generation procedure.

User Applications

In a CICS/DLI online environment, allocating PSBs or HD buffers and DL/I routines above the 16 MB line of storage is normally transparent to regular DL/I user applications.

If PSB copies reside above 16 MB (PSBLOC=ANY in ACT generation), consider the following:

- For application programs issuing the scheduling call in AMODE 31 using the DL/I call i/f, DL/I returns a PCB address list and PCB addresses located in 31-bit address space. Application programs scheduling the PSB in AMODE 24 using the DL/I call i/f will receive a PCB address list and PCB addresses located below 16 MB as before.
- The AMODE of scheduling the PSB has no effect for application programs implemented in the DL/I HLPI interface, because they do not access the PCB, the only part of the PSB that an application can use. HLPI programs will always receive a PSB copy in 31-bit address space.
- For application programs implemented in the DL/I call i/f that have issued the scheduling call in AMODE 31 - receiving a PCB address list and PCB addresses located above 16 MB - and then try to pass the PCB addresses to another program executing in AMODE 24, there is now a restriction:

  To make sure these programs are working correctly, it is necessary to either adapt the AMODEs of the applications to each other, or define PSB copies to be allocated below 16 MB (PSBLOC=GV24 or PSBLOC=BELOW in the ACT online generation).

In the DL/I batch environment, PSB and HD buffers are allocated below 16 MB. DL/I action and user exit routines will also reside below the line unless they have been loaded into the SVA (low or high). Application programs are expected to run unchanged.

User Exit Routines

In a CICS/DLI online environment, PSBs or HD buffers can reside above the 16 MB line (PSBLOC=ANY or HDMODE=ANY in ACT generation). In this case, DL/I calls the user exit routines in AMODE 31 regardless of their library attributes. In order to prevent possible errors, specifically of existing AMODE 24/RMODE 24 user exit routines now running in AMODE 31, you have to confirm in the ACT online generation by using the parameter USREXIT=AMODE-31 (as described in “Allocation of DL/I Resources above 16 MB” on page 58) that all exit routines used in the current DL/I installation are able to run in AMODE 31. In this case, it is not required to recreate existing user exit routines with AMODE 24 library attribute and build them as an AMODE 31 entry.

When there are exit routines that are not capable to run in AMODE 31, they have to be adapted for 31-bit execution, or – if this is not possible – you must define USREXIT=AMODE-24 in the ACT online generation. PSBs and HD buffers will then be allocated and parameters from DL/I 31-bit applications are copied below 16 MB. This way, all DL/I data resides below the line and DL/I calls the user exit routines in the addressing mode of their AMODE library attribute.

User exit routines in a DL/I batch environment are always executed in the addressing mode of their AMODE library attribute. PSB, HD buffers and all other DL/I data (including copied parameter lists from 31-bit applications) reside below 16 MB.

Note:

1. User exit routines referencing the PST control block must be reassembled and re-linkedited.
2. The linkage conventions for the user exit routines have not changed. Returning to DL/I should be done by BR R14.
3. User exit routines provided through the DL/I distribution libraries (DLZHDC10, DLZHDC20, DLZHDC30, DLZHDC40, DLZSAMCP) are delivered with the attributes AMODE 31/RMODE 24 or AMODE 31/RMODE ANY and can run in 31-bit addressing mode. That is, they would allow to set USREXIT=AMODE-31 in the ACT online generation.

4. When PSBs or HD buffers are residing below the 16 MB line, DL/I randomizing routines are called in AMODE 24.

Conditional SVA Loading

The SVA= parameter of the DLZACT TYPE=CONFIG macro has been disabled, because z/VSE 4.3 does not support the CICS coexistence environment anymore.

This parameter allowed DL/I VSE 1.11 running with CICS Transaction Server for VSE/ESA in a coexistence environment together with DL/I DOS/VS 1.10 and CICS/VSE 2.3 to use SVA-resident DL/I action modules from the SVA, or load them into CICS Transaction Server partition space and use them from there.

This is also described in “Conditional SVA Loading Disabled” on page 61 .

DL/I Trace Facility

All programs, which contain DLZTRACE macro calls must be reassembled and re-linkedited.

Migration Items from DL/I VSE 1.11 APARs

The following section describes the migration items resulting from APAR fixes for DL/I VSE 1.11 (before shipment of DL/I VSE 1.12.0).

Return UIB Address in User DIB at EXEC DLI

The following section describes the migration items resulting from APAR fixes for DL/I VSE 1.11 (before shipment of DL/I VSE 1.12.0) as described in “DL/I VSE 1.11 Enhancements” on page 61.

CICS/DLI online applications implemented in the HLPI (EXEC DLI) interface can pass the PCB address list of the PSB they have scheduled to another CICS/DLI online program implemented with the call-level interface through the field UIBPCBAL defined in the UIB control block. This is possible, because upon completion of a scheduling call, DL/I returns to HLPI-type programs the address of the UIB via the field DIBUIBA in the User DIB control block.

DL/I Support for LE/VSE UADUMP Option

When the LE/VSE UADUMP option has been set while at the same time DL/I abend handling with a partition dump was requested through UPSI bits 5 and 7 (set to zero), only one dump will be created when a DL/I batch application abnormally terminates: Either LE/VSE writes the partition dump and the DL/I abnormal termination routine suppresses the duplicate one, or, if LE/VSE was not able to take the dump, DL/I will do it instead.

Misleading Result Information in DLZMDLI0 for QURY Call

The result information from the QURY-function of the DL/I online test program DLZMDLI0 is now STARTED or STOPPED.

Scheduling Error when PSB is Generated with LANG=PLI

The restriction that PSBs of language-type PL/I are not supported for programs using the AIBTDLI interface has been removed. There is no more dependency between the language of the PSB and the language of the program when the AIBTDLI interface is used.

Chapter 9. Recovery after a DLZ004I or DLZ005I Error

When message DLZ004I (I/O-error) or DLZ005I (out-of-space) has occurred, the affected database(s) need to be recovered by a specific process before they can be used again. This process consists of the following steps:

1. Prepare the Log File.
2. Forward Recovery.
3. Backout of inflight changes.

To run these steps, the following DL/I batch utilities may be used:

- Log Print - DLZLOGP0.
- Data Set Recovery - DLZURDB0.
- Database Backout - DLZBACK0.

The description and a set of sample jobs of these utilities can be found in DL/I Resource Definition and Utilities. You may use these samples for the different recovery steps described below by adapting them to your environment and the specific task they should perform.

In a CICS/DLI online environment

CICS Emergency Restart

is needed for the backout of inflight changes, when 'DLIOER=ABEND' has been defined in the CICS SIT or CICS/DLI startup deck.

Input to the recovery process consists of the following sources:

- Backup (DL/I, VSAM, or other) of the dataset(s) of the database(s) to be recovered.
- Log File(s) and optionally DL/I Change Accumulation File covering the time from the backup to be used for recovery until the time, when DLZ004I / DLZ005I has occurred. This must include the Log File that was active at the time of the DLZ004I / DLZ005I error.

All recovery actions described below are based on the existence of these input sources. Otherwise, you have to recover the DLZ004I / DLZ005I database and, if needed, any database(s) related to it, by your own internal method (for example, via backups and repeating all work).

The recovery actions are shown as a step-by-step guideline adapted to the different situations, in which DLZ004I / DLZ005I can occur.

They also provide information, how a DL/I database can still be recovered when appropriate recovery was missing or not run correctly and the failing database(s) taken back into operation provided database backups and log data are available.

Before starting recovery, make sure PTF UI73493 is installed.

DLZ004I / DLZ005I in a CICS / DL/I Online Environment

If DLZ004I / DLZ005I has occurred in a CICS / DL/I online environment, the different recovery actions depend on, if 'DLIOER=ABEND' or 'DLIOER=CONTINUE' had been defined in the CICS SIT or CICS / DL/I startup deck.

In view of the expected effort, 'DLIOER=ABEND' probably allows DL/I recovery to be done easier and faster because CICS Emergency Restart can be used for the backout process. But CICS and DL/I are forced to terminate abnormally, and backout has to take place for all databases (with uncommitted changes) during CICS Emergency Restart, and not only the one that was affected by the DLZ004I / DLZ005I condition.

With 'DLIOER=CONTINUE', CICS and DL/I continue execution after the DLZ004I / DLZ005I error. DL/I stops the error database, which prevents that new CICS tasks can access this database or a database that is related to it. CICS and DL/I can later be terminated normally. Only the database(s) that were affected by the DLZ004I / DLZ005I condition have to be recovered.

Note: If after DLZ004I / DLZ005I you restarted CICS and DL/I without having taken appropriate recovery actions and see messages

DLZ141I	I/O ERROR HAS OCCURRED IN DATABASE dbdname
DLZ142A	ENTER CONTINUE (ONLY IF DATABASE RECOVERED), IGNORE OR CANCEL

enter

➢ **CANCEL**

to terminate CICS and DL/I initialization. This leaves your system open for the normal recovery path and you do not have to go to 'Recovering from missing or inappropriate DL/I Recovery', which may be more time consuming and roll back parts of the DL/I work you may have done in new CICS and DL/I cycles.

For a detailed description of the different responses that can be entered to message DLZ142A, refer to “Responses to Message DLZ142A” on page 92.

DLIOER=ABEND

When a DLZ004I / DLZ005I condition happened with DLIOER=ABEND defined, CICS additionally writes message DFHDL4540A and abnormally terminates the CICS and DL/I subsystem. DL/I recovery can then be run as shown below starting at “Prepare the CICS and DL/I Abnormal Termination Log File ” on page 70.

If your CICS/DLI system was running in a CICS XRF environment, refer to “Operating in a CICS XRF Environment” on page 72 for instructions on how to proceed.

In case you resumed CICS and DL/I operation without having taken appropriate DL/I recovery actions, “Recovering from missing or inappropriate DL/I Recovery” on page 73 provides information, how this is still possible afterward. But be aware that deferred recovery will be more time-consuming and may roll back parts of the DL/I work you may have done in new CICS and DL/I cycles.

Prepare the CICS and DL/I Abnormal Termination Log File

To ensure that the CICS log file has been correctly closed, you should restart CICS using the override parameter

START=LOGTERM.

This will automatically shut down CICS again.

Note: In a CICS XRF system you need to run this with XRF=NO defined.

DL/I Forward Recovery

From the name of the data set shown by message DLZ004I / DLZ005I, determine the name of the associated database that had failed and hence needs to be recovered. This is the name that was also shown by CICS message DHDL4540A, which followed DLZ004I / DLZ005I.

Note: Since at this point it's assumed - and expected - that before starting recovery, DL/I execution was stopped after the DLZ004I / DLZ005I CICS and DL/I abnormal termination for all databases that had been in use by the CICS/DLI online system, and the CICS/DLI online system remained shutdown:

When recovering the DLZ004I / DLZ005I error database, you solely have to recover the single data set shown by message DLZ004I / DLZ005I.

For this data set, run the DL/I data set recovery utility using this input:

- Backup (DL/I, VSAM, or other) of the data set.

- DL/I change accumulation file (if available) and all logs (batch and online) that had been created since the time of the backup to be used for recovery until and including the time of the DLZ004I / DLZ005I error, mounted in chronological order. Log files that had been condensed to and that are applied via a DL/I change accumulation file may not be included.

Backout Inflight Changes – CICS Emergency Restart

All databases are now in the state of the DLZ004I / DLZ005I abnormal termination. Because of that, uncommitted DL/I work may be existing that has now to be removed. In addition, all other resources, which may have been inflight at the time of the abnormal termination have to be recovered as well.

To do this, restart CICS and DL/I using

➢ START=AUTO to ensure CICS Emergency Restart is invoked, which resets all inflight CICS tasks and their resources,

and reply ➢ **CONTINUE** to messages

DLZ141I	I/O ERROR HAS OCCURRED IN DATABASE dbdname
DLZ142A	ENTER CONTINUE (ONLY IF DATABASE RECOVERED), IGNORE OR CANCEL

to backout all uncommitted DL/I database changes of DL/I tasks.

This completes recovery from the DLZ004I / DLZ005I error.

You may now want to repeat all work for the recovered databases originating from the data that had to be backed out. Then you can continue CICS/DLI online or resume DL/I batch operation.

For a detailed description of the different responses that can be entered to message DLZ142A, refer to “Responses to Message DLZ142A” on page 92.

DLIOER=CONTINUE

When a DLZ004I / DLZ005I condition happened with DLIOER=CONTINUE defined, CICS additionally writes message DFHDL4541A. DL/I stops the affected database, which prevents that new CICS tasks can access this database or a database that is related to it. The CICS task, which had caused DLZ004I / DLZ005I may or may not terminate abnormally – depending on, for example, the DL/I Programming Interface that was used. However, this makes no difference for the recovery process.

The CICS/DLI subsystem continues execution and can be terminated normally. Because of that, DL/I recovery cannot be run via CICS Emergency Restart, but requires DL/I batch utilities to be used as shown below in “DL/I Recovery” on page 72.

If CICS and DL/I should have terminated abnormally because of another failure that happened after the DLZ004I / DLZ005I error, CICS recovery is required as well. It has to be run first and separate from DL/I recovery as described in “CICS Recovery” on page 71 below.

If the CICS and DL/I abnormal termination has occurred while your CICS/DLI system was running in a CICS XRF environment, refer to “Operating in a CICS XRF Environment” on page 72 for instructions on how to proceed.

In case you resumed CICS and DL/I operation without having taken appropriate DL/I recovery actions, “Recovering from missing or inappropriate DL/I Recovery” on page 73 provides information, how this is still possible afterward. But be aware that deferred recovery may roll back parts of the DL/I work you may have done in new CICS and DL/I cycles.

CICS Recovery

When CICS has terminated abnormally, all CICS tasks that had been inflight at the time of the abnormal termination have to be recovered.

This is done by first restarting CICS using the override parameter.

- START=LOGTERM to ensure that the CICS log file has been correctly closed. This will automatically shut down CICS again.

  Note: In a CICS XRF system you need to run this with XRF=NO defined.

After that you should start CICS again with the following parameters:

- START=AUTO to ensure CICS Emergency Restart is invoked, which resets all inflight CICS tasks and their resources, and
- DLI=NO to prevent any DL/I activities / backout can take place at this time.

When CICS Emergency Restart has completed, all necessary CICS recovery has been done. Shut down CICS again and continue with "DL/I Recovery" on page 72 described below. Don't forget to reestablish your former DL/I startup parameter.

DL/I Recovery

To recover the databases that had been affected by a DLZ004I / DLZ005I error, follow the steps of "DL/I Batch Recovery Process II" on page 76.

If CICS and DL/I should have terminated abnormally because of another failure that happened after the DLZ004I / DLZ005I error, extended recovery actions as described in "DL/I Batch Recovery Process III" on page 79 are additionally needed.

Operating in a CICS XRF Environment

If your CICS/DLI system was running in a CICS XRF environment at the time of the CICS and DL/I abnormal termination, the standby CICS/DLI system will automatically be activated during XRF takeover and invoke CICS Emergency Restart. When you see DL/I messages

DLZ141I	I/O ERROR HAS OCCURRED IN DATABASE dbdname
DLZ142A	ENTER CONTINUE (ONLY IF DATABASE RECOVERED), IGNORE OR CANCEL

reply

➢ **CANCEL**

to terminate XRF takeover and allow recovery of the DLZ004I / DLZ005I database as shown starting at

- If CICS and DL/I abnormal termination occurred because of a DLZ004I / DLZ005I error with DLIOER=ABEND defined

  → "Prepare the CICS and DL/I Abnormal Termination Log File " on page 70.
- If CICS and DL/I terminated abnormally because of another failure that happened after the DLZ004I / DLZ005I error with DLIOER=CONTINUE defined

  → "CICS Recovery" on page 71.

For a detailed description of the different responses that can be entered to message DLZ142A, refer to "Responses to Message DLZ142A" on page 92.

In case you continued XRF takeover without having taken appropriate DL/I recovery actions, "Recovering from missing or inappropriate DL/I Recovery" on page 73 provides information, how this is still possible afterward. But be aware that deferred recovery may be more time consuming and roll back parts of the DL/I work you may have done in new CICS and DL/I cycles.

Recovering from missing or inappropriate DL/I Recovery

If after a DLZ004I / DLZ005I error in the CICS/DLI online environment you resumed CICS and DL/I operation without having taken appropriate recovery actions

- this includes having restarted CICS COLD -

you can run DL/I recovery at a later point of time as long as the necessary input sources are available. These are the database backups and the log data. If at the time of the DLZ004I / DLZ005I error

- 'DLIOER=ABEND' had been defined in the CICS SIT or CICS/DLI startup deck, you can run deferred DL/I recovery as described in "DL/I Batch Recovery Process I" on page 74.
- 'DLIOER=CONTINUE' had been defined in the CICS SIT or CICS/DLI startup deck, you can run deferred DL/I recovery as described in "DL/I Batch Recovery Process II" on page 76 – and additionally, if CICS and DL/I terminated abnormally because of another failure that happened after the DLZ004I / DLZ005I error, "DL/I Batch Recovery Process III" on page 79.

But be aware that deferred recovery may be more time-consuming and roll back parts of the DL/I work you may have done in new CICS and DL/I cycles.

DLZ004I / DLZ005I in a DL/I Batch Environment

If DLZ004I / DLZ005I has occurred in a DL/I batch environment, DL/I may terminate abnormally or continue execution, depending on, for example, the Programming Interface, in which the application was implemented. The recovery actions that can be taken, are shown below.

DL/I Batch Program Terminated Abnormally

When a DLZ004I / DLZ005I condition happened, which caused the DL/I batch program to terminate abnormally, DL/I additionally writes messages DLZ001I and DLZ002I. If this was the first/only incident of DLZ004I / DLZ005I in the job, DL/I recovery can be run as shown in "DL/I Batch Recovery Process IV" on page 82.

If before the abnormal termination (with or without a DLZ004I / DLZ005I condition) a preceding DLZ004I / DLZ005I error should have occurred with continued DL/I batch program execution, refer to "DL/I Batch Program Continued Execution" on page 73 below for instructions on how to proceed.

In case you resumed DL/I operation without having taken appropriate DL/I recovery actions, "Recovering from missing or inappropriate DL/I Recovery" on page 73 provides information, how this is still possible afterward. But be aware that deferred recovery will be more time-consuming and may roll back parts of the DL/I work you may have done in new DL/I cycles.

DL/I Batch Program Continued Execution

When a DLZ004I / DLZ005I condition happened and the DL/I batch program continued execution, DL/I had returned an I/O-error status code to the application, but didn't terminate abnormally. If no abnormal termination occurred after this in the job, DL/I recovery can be run by following the steps described in "DL/I Batch Recovery Process VI" on page 86.

If after a DLZ004I / DLZ005I error with continued DL/I batch program execution another failure - including a new DLZ004I / DLZ005I incident - should have occurred, which caused an abnormal termination now, extended recovery actions as described in "DL/I Batch Recovery Process VII" on page 88 are additionally needed.

In case you resumed DL/I operation without having taken appropriate DL/I recovery actions, "Recovering from missing or inappropriate DL/I Recovery" on page 73 provides information, how this is still possible afterward. But be aware that deferred recovery may roll back parts of the DL/I work you may have done in new DL/I cycles.

Recovering from missing or inappropriate DL/I Recovery

If after a DLZ004I / DLZ005I error in the DL/I batch environment you resumed DL/I operation without having taken appropriate recovery actions, you can run DL/I recovery at a later point of time as long as the necessary input sources are available. These are the database backups and the log data. If at the time of the first/only DLZ004I / DLZ005I error

- DL/I had terminated abnormally, you can run deferred DL/I recovery as described in “DL/I Batch Recovery Process V” on page 83.
- DL/I had continued execution, you can run deferred DL/I recovery as described in “DL/I Batch Recovery Process VI” on page 86 – and additionally, if another failure - including a new DLZ004I / DLZ005I incident - should have occurred, which caused an abnormal termination now, “DL/I Batch Recovery Process VII” on page 88.

But be aware that deferred recovery may be more time-consuming and roll back parts of the DL/I work you may have done in new DL/I cycles.

DL/I Batch Recovery Processes

The DL/I Batch Recovery Processes are run using DL/I batch utilities only. That is, also for a DLZ004I / DLZ005I error that had occurred in a CICS/DLI online environment they do not use CICS emergency restart for data backout.

The processes are similar in their structure and have been adapted to individual situations, in which DLZ004I / DLZ005I can occur.

“DL/I Batch Recovery Process I” on page 74, “DL/I Batch Recovery Process II” on page 76, and “DL/I Batch Recovery Process III” on page 79 are for use, when DLZ004I / DLZ005I has occurred during CICS/DLI online execution. “DL/I Batch Recovery Process IV” on page 82, “DL/I Batch Recovery Process V” on page 83, “DL/I Batch Recovery Process VI” on page 86, and “DL/I Batch Recovery Process VII” on page 88 are for use, when DLZ004I / DLZ005I has occurred during DL/I batch execution.

DL/I Batch Recovery Process I

At this point, it’s assumed – and mandatory – that, after the preceding DLZ004I / DLZ005I CICS and DL/I abnormal termination, the CICS/DL/I subsystem has now been terminated normally. Because of that, DL/I recovery from the DLZ004I / DLZ005I error can / may no more be run via CICS emergency restart, but requires DL/I batch utilities to be used as shown in steps 1. - 3. below.

1. Prepare the CICS and DL/I abnormal termination log file

   Create a tape-copy log file to be used for forward recovery and backout that ends with the last DL/I log record before the event of the DLZ004I / DLZ005I CICS and DL/I abnormal termination. To do this, run the DL/I Log print utility through steps 1a. to 1c. shown below:

   Note: Other than 'DLIOER=CONTINUE' does 'DLIOER=ABEND' not write an 'I/O error stop record' that could else be used to locate the event of the DLZ004I / DLZ005I CICS and DL/I abnormal termination on the log data set.

   a. Locate the '**L**ast **D**L/I log **R**ecord **B**efore **C**ICS and DL/I **A**bnormal **T**ermination' ('**LDRBCAT**') on the CICS journal and save its DL/I log sequence number.

      - Identify date & time of the CICS and DL/I abnormal termination using the timestamps provided in the console log.
      - Run the DL/I log print utility on all CICS journals that had been created during the CICS cycle, in which the CICS and DL/I abnormal termination had occurred, mounted in chronological order. Don’t pass any LO or LS input control statement to ensure all DL/I log records are printed. In the log print output, scan the 'DATE = ...' and 'TIME = ...' values of these records and find the last one that roughly coincides with the date & time of the CICS and DL/I abnormal termination as obtained from the console log. This could be a Data Base or Data Base / Backout record, or also a Scheduling or Termination record (but most likely not a Open or Stop record), of which the latter do not show a time stamp. The hereby located log record then represents the 'LDRBCAT' mentioned above. Save the 8-digit DL/I log sequence number from the keyword 'SEQNO =' of that record.

   b. Check, how often the DL/I log sequence number of the 'LDRBCAT' exists on the recovery logs. Save the number of that instance, which you counted at the 'LDRBCAT'.

Run the DL/I log print utility – no LO or LS input control statement – on all logs (batch and online) that had been created since the time of the backup to be used for recovery until and including the time of the CICS and DL/I abnormal termination, mounted in chronological order. This must include all CICS journals that had been written during the CICS/DLI cycle with the abnormal termination and that had also been used in step 1a. Log files that have been condensed to and that will later during forward recovery (step 2.) be applied via a DL/I change accumulation file may not be included. In the log print output, count all instances of a DL/I log sequence number that is identical to the one found with the 'LDRBCAT'. Save the number of that instance, which you counted at the 'LDRBCAT'.

c. Create a tape copy from the recovery logs that ends with the ‘LDRBCAT’.

Run the same log print job as used in step 1b. plus job control for the output tape and the following new parameters on the LO input control statement:

Column 41-44:	CPYR ... to request the tape-copy-by-record feature.
Column 54-61:	1 - 8-digit DL/I log sequence number of the 'LDRBCAT’ as found in step 1a., left-aligned.
Column 63-64:	1 - 2-digit version of the DL/I log sequence number of the 'LDRBCAT’ as found in step 1b., left-aligned. The default value is 1.

A detailed description of the new log print parameters can be found in “Log Print with Tape-Copy-by-Record (PM56161)” on page 7 and “Stop Log Print on Log Sequence Number” on page 5.

Note: For the creation of the tape-copy log file, you may use the VSE Virtual Tape facility.

✓ Save the tape copy created from the recovery logs.
✓ From the end of the log print output, save all pairs of messages DLZ424I and DLZ425I.

Below is an example of one DLZ424I / DLZ425I message pair:

DLZ424I	NO LOG TERMINATION RECORD FOUND FOR SCHEDULING RECORD
DLZ425I	PSBNAME = STBICL1P, SEQUENCE NUMBER = 00000044 DMBNAMEs = STDCDBPD, STDCY1PD, STDCX1PD, STDCX2PD

Each pair represents one inflight Logical Unit of Work (LUW) showing the name of a DL/I PSB that was in use – had been scheduled – by a CICS task at the time of the CICS and DL/I abnormal termination, together with the names of the databases – if any – that the LUW has changed.

Note: If – to run forward recovery at step 2. – all backups are to be used with the same set of log files (batch and online) – that is, all backups had been taken at around the same time – it’s sufficient to create one tape-copy log only. However, if backups correlate to different sets of log files, individual tape-copy logs have to be created through steps 1b. to 1c. for recovery of the individual databases. In this case, different log print executions at step 1c. should still show the same DLZ424I / DLZ425I message pairs.

2. DL/I forward recovery

The DLZ424I / DLZ425I message pairs in the log print output(s) obtained at step 1c., show all databases ('DMBNAMEs = ...') with inflight changes at the time of the CICS and DL/I abnormal termination. These databases - if any - constitute a – **initial** – list of databases to be recovered.

Note:

a. If the database of the DLZ004I / DLZ005I data set is not on the **initial** list, it has to be **added**.

b. Since in this recovery scenario it's assumed that before starting recovery, CICS and DL/I were executed again after the DLZ004I / DLZ005I CICS and DL/I abnormal termination:

When recovering a database, all related databases have to be recovered as well. This set of databases belonging together will then include

- The main database (ESDS).
- All index databases (primary and secondary indexes, KSDS) of the main database.
- All databases logically related to the main database and their index databases, and any databases logically related to them and their index databases, etc.

Of course – as an example – if the CICS and DL/I abnormal termination affected a HDAM database without indexes and logical relationship, the set contains one database only.

By adding the DLZ004I / DLZ005I error database (only if needed) and related databases, the mentioned initial list of databases to be recovered may become extended. This gives the – **final** – list of databases to be recovered.

c. You have to recover the full database. That is, if a database is defined on multiple data sets, you have to forward-recover each data set.

For the individual dataset(s) of the **final** list of database(s) to be recovered, run the DL/I data set recovery utility using this input:

- Backup (DL/I, VSAM, or other) of the data set.
- DL/I change accumulation file (if available) and → tape-copy log file created at step 1c.

3. Backout inflight changes

Having recovered the database(s) to the point of the CICS and DL/I abnormal termination, uncommitted work may be existing that has now to be removed. You do this by running the DL/I data base backout utility on those forward-recovered databases, for which inflight changes were pending at the time of the CICS and DL/I abnormal termination, that is:

Run the DL/I data base backout utility on each PSB that is shown by a DLZ424I / DLZ425I message pair together with inflight database changes ('DMBNAMEs = ...') in the log print output(s) obtained at step 1c. using the following input:

- Forward-recovered and unchanged databases that are accessed by the PSB, for which backout is executed.
- Tape-copy log file created at step 1c.

Note: If more than one tape-copy log file had to be created, you can now use any one of them.

Note:

- If databases with inflight changes were not existing, backout can / need not be executed.
- The new LS input control statement described in “Backout per Database in Batch” on page 6 must not to be entered here.
- You don't need to create an output log since new backups should be taken after the backout and a new log cycle started (see below).
- Before invoking the backout utility, you have to position the input log tape behind the log file (standard labeled). If the log file is the first file on the tape you may use this command:

```
// MTC FSF,cuu,2
```

cuu ... address of the tape device

This completes DL/I Batch Recovery Process I.

You may now want to repeat all work for the recovered databases originating from the data that had to be backed out.

You should also consider to “Start a new Log Cycle and Take new Backups” on page 91, then you can “Restart CICS and DL/I after DL/I Batch Recovery” on page 91 or resume DL/I batch operation.

DL/I Batch Recovery Process II

At this point, it’s assumed – and mandatory – that either:

- The CICS/DLI subsystem has been terminated normally, or
- All databases to be recovered have been individually closed online by using the DL/I STOP system call.

DL/I recovery from the DLZ004I / DLZ005I error can / may therefore not be run via CICS emergency restart, but requires DL/I batch utilities to be used as shown in steps 1. - 3. below.

Note: If DLZ004I or DLZ005I has occurred more than once during a CICS/DLI cycle, you need to run this process separately for each occurrence. For this reason, the 'I/O error log file' prepared at step 1. and used with forward recovery at step 2. and backout at step 3. has to be individually prepared for each DLZ004I / DLZ005I condition.
If databases, for which a DLZ004I / DLZ005I condition has been returned, are related to each other (via index or logical relationship), they can be recovered up to the **first** DLZ004I / DLZ005I incident that was shown for them only. In this case, the 'I/O error log file' prepared at step 1. and used with forward recovery at step 2. and backout at step 3. needs to be always the same and end with this first DLZ004I / DLZ005I incident for them on the log dataset(s).

1. Prepare the I/O error log file

   Create a tape-copy log file to be used for forward recovery and backout that ends with the DLZ004I / DLZ005I condition. This is represented by a 'I/O error stop record' on the log data set. To create the tape-copy log file, run the DL/I log print utility through steps 1a. to 1c. shown below:

   a. Locate the 'I/O error stop record' on the CICS journal and save its DL/I log sequence number.

      Run the DL/I log print utility on the CICS journal that was active at the time of the DLZ004I / DLZ005I error. Don't pass any LO or LS input control statement to ensure all DL/I log records are printed. In the log print output, search for the 'I/O error stop record' that was written with the DLZ004I or DLZ005I event.

      'I/O ERROR STOP RECORD, DSORG = ...'

      The name of the data set shown with message DLZ004I / DLZ005I must correspond to the name shown by the keyword 'FILENAME =' of this record.

      Save the 8-digit DL/I log sequence number from the keyword 'SEQNO ='.

   b. Check, how often the DL/I log sequence number of the 'I/O error stop record' exists on the recovery logs. Save the number of that instance, which you counted at the 'I/O error stop record'.

      Run the DL/I log print utility – no LO or LS input control statement – on all logs (batch and online) that had been created since the time of the backup to be used for recovery until and including the time of the DLZ004I / DLZ005I error, mounted in chronological order. This must include all CICS journals that had been written during the CICS/DLI cycle up to the DLZ004I / DLZ005I condition. Log files that have been condensed to and that will later during forward recovery (step 2.) be applied via a DL/I change accumulation file may not be included. In the log print output, count all instances of a DL/I log sequence number that is identical to the one found with the 'I/O error stop record'. Save the number of that instance, which you counted at the 'I/O error stop record'.

   c. Create a tape copy from the recovery logs that ends with the 'I/O error stop record'.

      Run the same log print job as used in step 1b. plus job control for the output tape and the following new parameters on the LO input control statement:

Column 41-44:	CPYR ... to request the tape-copy-by-record feature.
Column 54-61:	1 - 8-digit DL/I log sequence number of the 'I/O error stop record' as found in step 1a., left-aligned.
Column 63-64:	1 - 2-digit version of the DL/I log sequence number of the 'I/O error stop record' as found in step 1b., left-aligned. The default value is 1.

      A detailed description of the new log print parameters can be found in "Log Print with Tape-Copy-by-Record (PM56161)" on page 7 and "Stop Log Print on Log Sequence Number" on page 5.

      Note: For the creation of the tape-copy log file, you may use the VSE Virtual Tape facility.

✓ Save the tape copy created from the recovery logs.
✓ From the end of the log print output, save all pairs of messages DLZ424I and DLZ425I.

Below is an example of one DLZ424I / DLZ425I message pair:

DLZ424I	NO LOG TERMINATION RECORD FOUND FOR SCHEDULING RECORD
DLZ425I	PSBNAME = STBICL2P, SEQUENCE NUMBER = 00000088 DMBNAMEs = STDIDBPD, STDIY1PD, STDIX1PD

Each pair represents one inflight Logical Unit of Work (LUW) showing the name of a DL/I PSB that was in use – had been scheduled – by a CICS task at the time of the DLZ004I / DLZ005I error, together with the names of the databases – if any – that the LUW has changed.

Note: If – to run forward recovery at step 2. – all backups are to be used with the same set of log files (batch and online) – that is, all backups had been taken at around the same time – it's sufficient to create one tape-copy log only. However, if backups correlate to different sets of log files, individual tape-copy logs have to be created through steps 1b. to 1c. for recovery of the individual databases. In this case, different log print executions at step 1c. should still show the same DLZ424I / DLZ425I message pairs.

2. DL/I forward recovery

   From the name of the data set shown by message DLZ004I / DLZ005I, determine the name of the associated database that had failed and hence needs to be recovered. This is the name that was also shown by CICS message DFHDL4541A, which followed DLZ004I / DLZ005I.

   Note:

   a. Since in this recovery scenario it's assumed that before starting recovery, CICS and DL/I execution continued after the DLZ004I / DLZ005I condition:

      When recovering the DLZ004I / DLZ005I error database, all related databases have to be recovered as well. This set of databases belonging together will then include

      - The main database (ESDS).
      - All index databases (primary and secondary indexes, KSDS) of the main database.
      - All databases logically related to the main database and their index databases, and any databases logically related to them and their index databases, etc.

      Of course – as an example – if the error was at a HDAM database without indexes and logical relationship, the set contains one database only.

   b. You have to recover the full database. That is, if a database is defined on multiple data sets, you have to forward-recover each data set. The data set shown by message DLZ004I / DLZ005I may be only one of these data sets.

   For the individual dataset(s) of the database(s) to be recovered, run the DL/I data set recovery utility using this input:

   - Backup (DL/I, VSAM, or other) of the data set.
   - DL/I change accumulation file (if available) and → tape-copy log file created at step 1c.

3. Backout inflight changes

   Having recovered the database(s) to the point of the DLZ004I / DLZ005I error, uncommitted work may be existing that has now to be removed. You do this by running the DL/I data base backout utility on those forward-recovered databases, for which inflight changes were pending at the time of the DLZ004I / DLZ005I error. To identify these databases and the PSBs, which used them, proceed as follows:

   - Create a list of all databases that had been forward-recovered at step 2. As mentioned there, this list may also contain one database only.
   - For each of the inflight PSBs shown by DLZ424I / DLZ425I message pairs in the log print output(s) obtained at step 1c. determine, if there are database names shown in message DLZ425I via

'DMBNAMEs = ...' that coincide with a database name from the list of forward-recovered databases. For each match, this database has to be backed out for this PSB.

Note: If no match is found for a PSB, or the PSB doesn't show any database(s) with inflight changes, backout can / need not be executed for this PSB.

To remove inflight changes from the forward-recovered databases, run the DL/I data base backout utility on each PSB that the steps above found as having inflight changes with one or more of the forward-recovered databases, using the following input:

- Forward-recovered and unchanged databases that are accessed by the PSB, for which backout is executed.
- Tape-copy log file created at step 1c.

  Note: If more than one tape-copy log file had to be created, you can now use any one of them.
- One new LS input control statement for each forward-recovered database, which has been found as to have inflight changes with the PSB, for which backout is executed:

Column 1-2:	LS ... to identify the input control statement.
Column 4-10:	Name of the database to be backed out.

A detailed description of the new LS input control statement can be found in “Backout per Database in Batch” on page 6.

Note:

- It’s mandatory that you supply LS input control statements for the individual databases to be backed out. Otherwise, non-forward-recovered databases that had not been affected by the DLZ004I / DLZ005I error might inadvertently be backed out as well and get corrupted.
- You don't need to create an output log since new backups should be taken after the backout and a new log cycle started (see below).
- Before invoking the backout utility, you have to position the input log tape behind the log file (standard labeled). If the log file is the first file on the tape you may use this command:

```
// MTC FSF,cuu,2
```

cuu ... address of the tape device

This completes DL/I Batch Recovery Process II.

You may now want to repeat all work for the recovered databases originating from the time after the DLZ004I / DLZ005I error and the data that had to be backed out.

When CICS and DL/I terminated normally after the DLZ004I / DLZ005I error, you should also consider to “Start a new Log Cycle and Take new Backups” on page 91, then you can “Restart CICS and DL/I after DL/I Batch Recovery” on page 91 or resume DL/I batch operation.

In case CICS and DL/I were up and running during the recovery process, you can now restart the recovered databases by using the DL/I STRT system call and continue CICS and DL/I operation with them. Additionally, you should consider to “Start a new Log Cycle and Take new Backups” on page 91, then you can “Restart CICS and DL/I after DL/I Batch Recovery” on page 91 or resume DL/I batch operation.

If CICS and DL/I should have terminated abnormally because of another failure that happened after the DLZ004I / DLZ005I error, you have to continue with extended recovery actions as described in “DL/I Batch Recovery Process III” on page 79 below.

DL/I Batch Recovery Process III

This describes the extended recovery actions needed in addition to the recovery actions described above at “DL/I Batch Recovery Process II” on page 76 when CICS and DL/I had terminated abnormally because of another failure that happened after the DLZ004I / DLZ005I error. They have to be taken to recover those databases that were impacted by the CICS and DL/I abnormal termination, but had not been

recovered in “DL/I Batch Recovery Process II” on page 76 dedicatedly as they were not affected by a DLZ004I / DLZ005I situation.

To recover these databases, follow steps 1. - 3. below:

1. Prepare the CICS and DL/I abnormal termination log file

   Create a tape-copy log file to be used for forward recovery and backout that ends with the last DL/I log record before the event of the CICS and DL/I abnormal termination. To do this, run the DL/I log print utility through steps 1a. to 1c. shown below:

   a. Locate the '**L**ast **D**L/I log **R**ecord **B**efore **C**ICS and DL/I **A**bnormal **T**ermination' (‘LDRBCAT’) on the CICS journal and save its DL/I log sequence number.

      - Identify date and time of the CICS and DL/I abnormal termination using the timestamps provided in the console log.
      - Run the DL/I log print utility on all CICS journals that had been created during the CICS cycle, in which the CICS and DL/I abnormal termination had occurred, mounted in chronological order. Don’t pass any LO or LS input control statement to ensure all DL/I log records are printed. In the log print output, scan the 'DATE = ...' and 'TIME = ...' values of these records and find the last one that roughly coincides with the date & time of the CICS and DL/I abnormal termination as obtained from the console log. This could be a Data Base or Data Base / Backout record, or also a Scheduling or Termination record (but most likely not a Open or Stop record), of which the latter do not show a time stamp. The hereby located log record then represents the ‘LDRBCAT’ mentioned above. Save the 8-digit DL/I log sequence number from the keyword 'SEQNO =' of that record.

   b. Check, how often the DL/I log sequence number of the ‘LDRBCAT’ exists on the recovery logs. Save the number of that instance, which you counted at the ‘LDRBCAT’.

      Run the DL/I log print utility – no LO or LS input control statement – on all logs (batch and online) that had been created since the time of the backup to be used for recovery until and including the time of the CICS and DL/I abnormal termination, mounted in chronological order. This must include all CICS journals that had been written during the CICS/DLI cycle with the abnormal termination and that had also been used in step 1a. Log files that have been condensed to and that will later during forward recovery (step 2.) be applied via a DL/I change accumulation file may not be included. In the log print output, count all instances of a DL/I log sequence number that is identical to the one found with the ‘LDRBCAT’. Save the number of that instance, which you counted at the ‘LDRBCAT’.

   c. Create a tape copy from the recovery logs that ends with the ‘LDRBCAT’.

      Run the same log print job as used in step 1b. plus job control for the output tape and the following new parameters on the LO input control statement:

Column 41-44:	CPYR ... to request the tape-copy-by-record feature.
Column 54-61:	1 - 8-digit DL/I log sequence number of the 'LDRBCAT’ as found in step 1a., left-aligned.
Column 63-64:	1 - 2-digit version of the DL/I log sequence number of the 'LDRBCAT’ as found in step 1b., left-aligned. The default value is 1.

      A detailed description of the new log print parameters can be found in “Log Print with Tape-Copy-by-Record (PM56161)” on page 7 and “Stop Log Print on Log Sequence Number” on page 5.

      Note: For the creation of the tape-copy log file, you may use the VSE Virtual Tape facility.

      ✓ Save the tape copy created from the recovery logs.
      ✓ From the end of the log print output, save all pairs of messages DLZ424I and DLZ425I.

      Below is an example of one DLZ424I / DLZ425I message pair:

DLZ424I	NO LOG TERMINATION RECORD FOUND FOR SCHEDULING RECORD

DLZ425I	PSBNAME = STBICL3P, SEQUENCE NUMBER = 00000132 DMBNAMEs = STDCDBPD, STDCY1PD, STDCX1PD,STDCX2PD, STDIDBPD, STDIY1PD, STDIX1PD

Each pair represents one inflight Logical Unit of Work (LUW) showing the name of a DL/I PSB that was in use – had been scheduled – by a CICS task at the time of the CICS and DL/I abnormal termination, together with the names of the databases – if any – that the LUW has changed.

Note: If – to run forward recovery at step 2. – all backups are to be used with the same set of log files (batch and online) – that is, all backups had been taken at around the same time – it's sufficient to create one tape-copy log only. However, if backups correlate to different sets of log files, individual tape-copy logs have to be created through steps 1b. to step 1c. for recovery of the individual databases. In this case, different log print executions at step 1c. should still show the same DLZ424I / DLZ425I message pairs.

2. DL/I forward recovery

   The DLZ424I / DLZ425I message pairs in the log print output(s) obtained at step 1c., show all databases ('DMBNAMEs = ...') with inflight changes at the time of the CICS and DL/I abnormal termination. These databases - if any - constitute a – **initial** – list of databases to be recovered.

   Note: If databases with inflight changes are not existing, forward recovery (and backout) can / need not be executed.

   a. Since in this recovery scenario it could be the case that before starting recovery, CICS and DL/I were executed again after the CICS and DL/I abnormal termination:

      When recovering a database, all related databases have to be recovered as well. This set of databases belonging together will then include

      - The main database (ESDS).
      - All index databases (primary and secondary indexes, KSDS) of the main database.
      - All databases logically related to the main database and their index databases, and any databases logically related to them and their index databases, etc.

      Of course – as an example – if the CICS and DL/I abnormal termination affected a HDAM database without indexes and logical relationship, the set contains one database only.

      By adding related databases, the mentioned initial list of databases to be recovered may become **extended.**

   b. Databases of the – now possibly **extended** – list of databases to be recovered, which have already been recovered for a DLZ004I / DLZ005I situation by "DL/I Batch Recovery Process II" on page 76 above, have to be **removed** from the list. This gives the – **final** – list of databases to be recovered.

   c. You have to recover the full database. That is, if a database is defined on multiple data sets, you have to forward-recover each data set.

   For the individual dataset(s) of the **final** list of database(s) to be recovered, run the DL/I data set recovery utility using this input:

   - Backup (DL/I, VSAM, or other) of the data set.
   - DL/I change accumulation file (if available) and → tape-copy log file created at step 1c.

3. Backout inflight changes

   Having recovered the database(s) to the point of the CICS and DL/I abnormal termination, uncommitted work may be existing that has now to be removed. You do this by running the DL/I data base backout utility on those databases, for which inflight changes were pending at the time of the CICS and DL/I abnormal termination, and that had not already been recovered for a DLZ004I / DLZ005I situation in "DL/I Batch Recovery Process II" on page 76. To identify these databases and the PSBs, which used them, proceed as follows:

   - For each of the inflight PSBs shown by DLZ424I / DLZ425I message pairs in the log print output(s) obtained at step 1c., determine, if there are database names shown in message DLZ425I via

'DMBNAMEs = ...' that coincide with a database name from the list of forward-recovered databases. For each match, this database has to be backed out for this PSB.

Note: If no match is found for a PSB, or the PSB doesn't show any database(s) with inflight changes, backout can / need not be executed for this PSB.

To remove inflight changes from the forward-recovered databases, run the DL/I data base backout utility on each PSB that the steps above found as having inflight changes to one or more of the forward-recovered databases, using the following input:

- Forward-recovered and unchanged databases that are accessed by the PSB, for which backout is executed.
- Tape-copy log file created at step 1c.

  Note: If more than one tape-copy log file had to be created, you can now use any one of them.
- One new LS input control statement for each forward-recovered database, which has been found as to have inflight changes with the PSB, for which backout is executed:

Column 1-2:	LS ... to identify the input control statement.
Column 4-10:	Name of the database to be backed out.

A detailed description of the new LS input control statement can be found in “Backout per Database in Batch” on page 6.

Note:

- It’s mandatory that you supply LS input control statements for the individual databases to be backed out. Otherwise, non-forward-recovered databases that may have been excluded at step 2b. might inadvertently be backed out as well and get corrupted.
- You don't need to create an output log since new backups should be taken after the backout and a new log cycle started (see below).
- Before invoking the backout utility, you have to position the input log tape behind the log file (standard labeled). If the log file is the first file on the tape you may use this command:

```
// MTC FSF,cuu,2
```

cuu ... address of the tape device

This completes DL/I Batch Recovery Process III.

You may now want to repeat all work for the recovered databases originating from the data that had to be backed out.

You should also consider to “Start a new Log Cycle and Take new Backups” on page 91, then you can “Restart CICS and DL/I after DL/I Batch Recovery” on page 91 or resume DL/I batch operation.

DL/I Batch Recovery Process IV

Follow steps 1. - 3. below:

1. Prepare the DL/I abnormal termination log file

   Create a tape-copy log file to be used for forward recovery and backout that ends with the last DL/I log record on the log file that was active at the time of the DL/I abnormal termination. To do this

   - Run the DL/I log print utility on all logs (batch and online) that had been created since the time of the backup to be used for recovery until and including the time of the DL/I abnormal termination, mounted in chronological order. This must include all log data sets that had been written during the batch job up to the DLZ004I / DLZ005I condition. Log files that have been condensed to and that will later during forward recovery (step 2.) be applied via a DL/I change accumulation file may not be included.

     Use these definitions:

– On the LO input control statement, column 41-44, enter

COPY ... to request the tape-copy-feature.

– No LS input control statement.

– Add job control for the output tape.

Note: For the creation of the tape-copy log file you may use the VSE Virtual Tape facility.

✓ Save the tape copy created from the recovery logs.

2. DL/I forward recovery

From the name of the data set shown by message DLZ004I / DLZ005I, determine the name of the associated database that had failed and hence needs to be recovered.

Note: Since in this recovery scenario it's assumed - and expected - that before starting recovery, DL/I execution was stopped after the DLZ004I / DLZ005I abnormal termination for all databases that had been in use by the failing DL/I batch job:

When recovering the DLZ004I / DLZ005I error database, you solely have to recover the single data set shown by message DLZ004I / DLZ005I.

For this data set, run the DL/I data set recovery utility using this input:

- Backup (DL/I, VSAM, or other) of the data set
- DL/I change accumulation file (if available) and → tape-copy log file created above.

3. Backout inflight changes

All databases are now in the state of the DLZ004I / DLZ005I abnormal termination. Because of that, uncommitted DL/I work may be existing that has now to be removed. To do this, run the DL/I data base backout utility

→ with DL/I batch logging being active

on the PSB of the execution that had failed using the following input:

- Forward-recovered and unchanged databases that are accessed by the PSB, for which backout is executed.
- Tape-copy log file created above.

Note:

- The new LS input control statement described in “Backout per Database in Batch” on page 6 must not to be entered here.
- Before invoking the backout utility, you have to position the input log tape behind the log file (standard labeled). If the log file is the first file on the tape you may use this command:

```
// MTC FSF,cuu,2
```

cuu ... address of the tape device.

Save the log file that has been created during the backout process for potential future recovery actions.

This completes DL/Batch Recovery Process IV.

You may now want to repeat all work for the recovered databases originating from the data that had to be backed out.

Then you can resume DL/I batch or CICS/DLI online operation.

DL/I Batch Recovery Process V

Follow steps 1. - 3. below:

1. Prepare the DL/I abnormal termination log file.

Create a tape-copy log file to be used for forward recovery and backout that ends with the last DL/I log record on the log file that was active at the time of the DL/I abnormal termination. To do this, run the DL/I log print utility through steps 1a. to 1c., shown below:

a. Locate the '**L**ast **D**L/I log **R**ecord before DL/I **B**atch **A**bnormal **T**ermination' ('LDRBAT') on the log data set and save its DL/I log sequence number.

   - Run the DL/I log print utility on the log file that was active at the time of the DL/I abnormal termination. Don't pass any LO or LS input control statement to ensure all log records are printed. In the log print output, find the last one of these records. It represents the 'LDRBAT' mentioned above. Save the 8-digit DL/I log sequence number from the keyword 'SEQNO =' of that record.

b. Check, how often the DL/I log sequence number of the 'LDRBAT' exists on the recovery logs. Save the number of that instance, which you counted at the 'LDRBAT'.

   Run the DL/I log print utility – no LO or LS input control statement – on all logs (batch and online) that had been created since the time of the backup to be used for recovery until and including the time of the DL/I abnormal termination, mounted in chronological order. This must include all log data sets that had been written during the batch job up to the DL/I abnormal termination. Log files that have been condensed to and that will later during forward recovery (step 2.) be applied via a DL/I change accumulation file may not be included. In the log print output, count all instances of a DL/I log sequence number that is identical to the one found with the 'LDRBAT'. Save the number of that instance, which you counted at the 'LDRBAT'.

c. Create a tape copy from the recovery logs that ends with the 'LDRBAT'.

   Run the same log print job as used in step 1b. plus job control for the output tape and the following new parameters on the LO input control statement:

Column 41-44:	CPYR ... to request the tape-copy-by-record feature.
Column 54-61:	1 - 8-digit DL/I log sequence number of the 'LDRBAT' as found in step 1a., left-aligned.
Column 63-64:	1 - 2-digit version of the DL/I log sequence number of the 'LDRBAT' as found in step 1b., left-aligned. The default value is 1.

   A detailed description of the new log print parameters can be found in "Log Print with Tape-Copy-by-Record (PM56161)" on page 7 and "Stop Log Print on Log Sequence Number" on page 5.

   Note: For the creation of the tape-copy log file, you may use the VSE Virtual Tape facility.

   ✓ Save the tape copy created from the recovery logs.
   ✓ From the end of the log print output, save the message pair DLZ424I / DLZ425I.

   Below is an example of a DLZ424I / DLZ425I message pair:

DLZ424I	NO LOG TERMINATION RECORD FOUND FOR SCHEDULING RECORD
DLZ425I	PSBNAME = STBICL4P, SEQUENCE NUMBER = 00000176 DMBNAMEs = STDCDBPD, STDIDBPD, STDCY1PD, STDCX1PD

   The message pair shows the name of the DL/I PSB that was in use at the time of the DL/I batch abnormal termination, together with the names of the databases – if any – the application has changed since program start or the last DL/I checkpoint.

   Note: If – to run forward recovery at step 2. – all backups are to be used with the same set of log files (batch and online) – that is, all backups had been taken at around the same time – it's sufficient to create one tape-copy log only. However, if backups correlate to different sets of log files, individual tape-copy logs have to be created through steps 1b. to step 1c. for recovery of the individual databases. In this case, different log print executions at step 1c. should still show the same DLZ424I / DLZ425I message pair.

2. DL/I forward recovery

The DLZ424I / DLZ425I message pair in the log print output(s) obtained at step 1c., shows all databases ('DMBNAMEs = ...') with inflight changes at the time of the DL/I abnormal termination. These databases - if any - constitute a – **initial** – list of databases to be recovered.

Note:

a. If the database of the DLZ004I / DLZ005I data set is not on the **initial** list, it has to be **added**.

b. Since in this recovery scenario it's assumed that before starting recovery, DL/I was executed again after the DL/I abnormal termination:

   When recovering a database, all related databases have to be recovered as well. This set of databases belonging together will then include

   - The main database (ESDS).
   - All index databases (primary and secondary indexes, KSDS) of the main database.
   - All databases logically related to the main database and their index databases, and any databases logically related to them and their index databases, etc.

   Of course – as an example – if the DL/I abnormal termination affected a HDAM database without indexes and logical relationship, the set contains one database only.

   By adding the DLZ004I / DLZ005I error database (only if needed) and related databases, the mentioned initial list of databases to be recovered may become extended. This gives the – **final** – list of databases to be recovered.

c. You have to recover the full database. That is, if a database is defined on multiple data sets, you have to forward-recover each data set.

   For the individual dataset(s) of the **final** list of database(s) to be recovered, run the DL/I data set recovery utility using this input:

   - Backup (DL/I, VSAM, or other) of the data set.
   - DL/I change accumulation file (if available) and → tape-copy log file created at step 1c.

3. Backout inflight changes

Having recovered the database(s) to the point of the DL/I abnormal termination, uncommitted work may be existing that has now to be removed. You do this by running the DL/I data base backout utility on those forward-recovered databases, for which inflight changes were pending at the time of the DL/I abnormal termination, that is:

Run the DL/I data base backout utility on the PSB that was active at the time of the DL/I abnormal termination using the following input:

- Forward-recovered and unchanged databases that are accessed by the PSB, for which backout is executed.
- Tape-copy log file created at step 1c.

  Note: If more than one tape-copy log file had to be created, you can now use any one of them.

Note:

- If databases with inflight changes were not existing, backout can / need not be executed.
- The new LS input control statement described in “Backout per Database in Batch” on page 6 must not to be entered here.
- You don't need to create an output log since new backups should be taken after the backout and a new log cycle started (see below).
- Before invoking the backout utility, you have to position the input log tape behind the log file (standard labeled). If the log file is the first file on the tape you may use this command:

```
// MTC FSF,cuu,2
```

  cuu ... address of the tape device

This completes DL/I Batch Recovery Process V.

You may now want to repeat all work for the recovered databases originating from the data that had to be backed out.

You should also consider to “Start a new Log Cycle and Take new Backups” on page 91, then you can resume DL/I batch or CICS/DLI online operation.

DL/I Batch Recovery Process VI

Note: If DLZ004I or DLZ005I has occurred more than once during DL/I batch execution, you need to run this process separately for each occurrence. For this reason, the 'I/O error log file' prepared at step 1. and used with forward recovery at step 2. and backout at step 3. has to be individually prepared for each DLZ004I / DLZ005I condition.
If databases, for which a DLZ004I / DLZ005I condition has been returned, are related to each other (via index or logical relationship), they can be recovered up to the **first** DLZ004I / DLZ005I incident that was shown for them only. In this case, the 'I/O error log file' prepared at step 1. and used with forward recovery at step 2. and backout at step 3. needs to be always the same and end with this first DLZ004I / DLZ005I incident for them on the log dataset(s).

Follow steps 1. - 3. below:

1. Prepare the I/O error log file.

   Create a tape-copy log file to be used for forward recovery and backout that ends with the DLZ004I / DLZ005I condition. This is represented by a 'I/O error stop record' on the log data set. To create the tape-copy log file, run the DL/I log print utility through steps 1a. to 1c. shown below:

   a. Locate the 'I/O error stop record' on the log data set and save its DL/I log sequence number.

      Run the DL/I log print utility on the log file that was active at the time of the DLZ004I / DLZ005I error. Don't pass any LO or LS input control statement to ensure all log records are printed. In the log print output, search for the 'I/O error stop record' that was written with the DLZ004I or DLZ005I event:

      'I/O ERROR STOP RECORD, DSORG = ...'

      The name of the data set shown with message DLZ004I / DLZ005I must correspond to the name shown by the keyword 'FILENAME =' of this record.

      Save the 8-digit DL/I log sequence number from the keyword 'SEQNO ='.

   b. Check, how often the DL/I log sequence number of the 'I/O error stop record' exists on the recovery logs. Save the number of that instance, which you counted at the 'I/O error stop record'.

      Run the DL/I log print utility – no LO or LS input control statement – on all logs (batch and online) that had been created since the time of the backup to be used for recovery until and including the time of the DLZ004I / DLZ005I error, mounted in chronological order. This must include all log data sets that had been written during the batch job up to the DLZ004I / DLZ005I condition. Log files that have been condensed to and that will later during forward recovery (step 2.) be applied via a DL/I change accumulation file may not be included. In the log print output, count all instances of a DL/I log sequence number that is identical to the one found with the 'I/O error stop record'. Save the number of that instance, which you counted at the 'I/O error stop record'.

   c. Create a tape copy from the recovery logs that ends with the 'I/O error stop record'.

      Run the same log print job as used in step 1b. plus job control for the output tape and the following new parameters on the LO input control statement:

Column 41-44:	CPYR ... to request the tape-copy-by-record feature.
Column 54-61:	1 - 8-digit DL/I log sequence number of the 'I/O error stop record' as found in step 1a., left-aligned.
Column 63-64:	1 - 2-digit version of the DL/I log sequence number of the 'I/O error stop record' as found in step 1b., left-aligned. The default value is 1.

A detailed description of the new log print parameters can be found in “Log Print with Tape-Copy-by-Record (PM56161)” on page 7 and “Stop Log Print on Log Sequence Number” on page 5.

Note: For the creation of the tape-copy log file, you may use the VSE Virtual Tape facility.

✓ Save the tape copy created from the recovery logs.
✓ From the end of the log print output, save the message pair DLZ424I / DLZ425I.

Below is an example of a DLZ424I / DLZ425I message pair:

DLZ424I	NO LOG TERMINATION RECORD FOUND FOR SCHEDULING RECORD
DLZ425I	PSBNAME = STBICL5P, SEQUENCE NUMBER = 00000220 DMBNAMEs = STDCDBPD, STDIDBPD, STDCY1PD, STDCX1PD, STDCX2PD

The message pair shows the name of the DL/I PSB that was in use at the time of the DLZ004I / DLZ005I error, together with the names of the databases – if any – the application has changed since program start or the last DL/I checkpoint.

Note: If – to run forward recovery at step 2. – all backups are to be used with the same set of log files (batch and online) – that is, all backups had been taken at around the same time – it's sufficient to create one tape-copy log only. However, if backups correlate to different sets of log files, individual tape-copy logs have to be created through steps 1b. to 1c. for recovery of the individual databases. In this case, different log print executions at step 1c. should still show the same DLZ424I / DLZ425I message pair.

2. DL/I forward recovery

   From the name of the data set shown by message DLZ004I / DLZ005I, determine the name of the associated database that had failed and hence needs to be recovered.

   Note:

   a. Since in this recovery scenario it's assumed that before starting recovery, DL/I execution continued after the DLZ004I / DLZ005I condition:

      When recovering the DLZ004I / DLZ005I error database, all related databases have to be recovered as well. This set of databases belonging together will then include

      - The main database (ESDS).
      - All index databases (primary and secondary indexes, KSDS) of the main database.
      - All databases logically related to the main database and their index databases, and any databases logically related to them and their index databases, etc.

      Of course – as an example – if the error was at a HDAM database without indexes and logical relationship, the set contains one database only.

   b. You have to recover the full database. That is, if a database is defined on multiple data sets, you have to forward-recover each data set. The data set shown by message DLZ004I / DLZ005I may be only one of these data sets.

      For the individual dataset(s) of the database(s) to be recovered, run the DL/I data set recovery utility using this input:

      - Backup (DL/I, VSAM, or other) of the data set.
      - DL/I change accumulation file (if available) and → tape-copy log file created at step 1c.

3. Backout inflight changes.

   Having recovered the database(s) to the point of the DLZ004I / DLZ005I error, uncommitted work may be existing that has now to be removed. You do this by running the DL/I data base backout utility on those forward-recovered databases, for which inflight changes were pending at the time of the DLZ004I / DLZ005I error. To identify these databases, proceed as follows:

   - Create a list of all databases that had been forward-recovered at step 2. As mentioned there, this list may also contain one database only.

- For the inflight PSB shown by the DLZ424I / DLZ425I message pair in the log print output(s) obtained at step 1c., determine, if there are database names shown in message DLZ425I via 'DMBNAMEs = ...' that coincide with a database name from the list of forward-recovered databases. For each match, this database has to be backed out for this PSB.

  Note: If no match is found for the PSB, or the PSB doesn't show any database(s) with inflight changes, backout can / need not be executed for this PSB.

To remove inflight changes from the forward-recovered databases, run the DL/I data base backout utility on the PSB that was active at the time of the DLZ004I / DLZ005I error using the following input:

- Forward-recovered and unchanged databases that are accessed by the PSB, for which backout is executed.
- Tape-copy log file created at step 1c.

  Note: If more than one tape-copy log file had to be created, you can now use any one of them.
- One new LS input control statement for each forward-recovered database, which has been found as to have inflight changes with the PSB, for which backout is executed:

Column 1-2:	LS ... to identify the input control statement.
Column 4-10:	Name of the database to be backed out.

A detailed description of the new LS input control statement can be found in “Backout per Database in Batch” on page 6.

Note:

- It's mandatory that you supply LS input control statements for the individual databases to be backed out. Otherwise, non-forward-recovered databases that had not been affected by the DLZ004I / DLZ005I error might inadvertently be backed out as well and get corrupted.
- You don't need to create an output log since new backups should be taken after the backout and a new log cycle started (see below).
- Before invoking the backout utility, you have to position the input log tape behind the log file (standard labeled). If the log file is the first file on the tape you may use this command:

```
// MTC FSF,cuu,2
```

cuu ... address of the tape device

This completes DL/I Batch Recovery Process VI.

You may now want to repeat all work for the recovered databases originating from the time after the DLZ004I / DLZ005I error and the data that had to be backed out.

You should also consider to “Start a new Log Cycle and Take new Backups” on page 91, then you can resume DL/I batch or CICS/DLI online operation.

If DL/I batch execution should have terminated abnormally because of another failure that happened after the DLZ004I / DLZ005I error, or because DLZ004I / DLZ005I had caused an abnormal termination now, extended recovery actions as described below in “DL/I Batch Recovery Process VII” on page 88 are additionally needed.

DL/I Batch Recovery Process VII

This describes the extended recovery actions needed in addition to the recovery actions described above at “DL/I Batch Recovery Process VI” on page 86, when DL/I batch execution had terminated abnormally because of another failure that happened after the DLZ004I / DLZ005I error, or because DLZ004I / DLZ005I had caused an abnormal termination now. They have to be taken to recover those databases that were impacted by the DL/I abnormal termination, but had not been recovered in “DL/I Batch Recovery Process VI” on page 86 dedicatedly as they were not affected by a DLZ004I / DLZ005I situation after which the DL/I batch program continued execution.

To recover these databases, follow steps 1. - 3. below:

1. Prepare the DL/I abnormal termination log file

   Create a tape-copy log file to be used for forward recovery and backout that ends with the last DL/I log record on the log file that was active at the time of the DL/I abnormal termination. To do this, run the DL/I log print utility through steps 1a. to 1c. shown below:

   a. Locate the '**L**ast **D**L/I log **R**ecord before DL/I **B**atch **A**bnormal **T**ermination' (‘LDRBAT’) on the log data set and save its DL/I log sequence number.

      - Run the DL/I log print utility on the log file that was active at the time of the DL/I abnormal termination. Don't pass any LO or LS input control statement to ensure all log records are printed. In the log print output, find the last one of these records. It represents the ‘LDRBAT’ mentioned above. Save the 8-digit DL/I log sequence number from the keyword 'SEQNO =' of that record.

   b. Check, how often the DL/I log sequence number of the ‘LDRBAT’ exists on the recovery logs. Save the number of that instance, which you counted at the ‘LDRBAT’.

      Run the DL/I log print utility – no LO or LS input control statement – on all logs (batch and online) that had been created since the time of the backup to be used for recovery until and including the time of the DL/I abnormal termination, mounted in chronological order. This must include all log data sets that had been written during the batch job up to the DL/I abnormal termination. Log files that have been condensed to and that will later during forward recovery (step 2.) be applied via a DL/I change accumulation file may not be included. In the log print output, count all instances of a DL/I log sequence number that is identical to the one found with the ‘LDRBAT’. Save the number of that instance, which you counted at the ‘LDRBAT’.

   c. Create a tape copy from the recovery logs that ends with the ‘LDRBAT’.

      Run the same log print job as used in step 1b. plus job control for the output tape and the following new parameters on the LO input control statement:

Column 41-44:	CPYR ... to request the tape-copy-by-record feature.
Column 54-61:	1 - 8-digit DL/I log sequence number of the 'LDRBAT’ as found in step 1a., left-aligned.
Column 63-64:	1 - 2-digit version of the DL/I log sequence number of the 'LDRBAT’ as found in step 1b., left-aligned. The default value is 1.

      A detailed description of the new log print parameters can be found in “Log Print with Tape-Copy-by-Record (PM56161)” on page 7 and “Stop Log Print on Log Sequence Number” on page 5.

      Note: For the creation of the tape-copy log file, you may use the VSE Virtual Tape facility.

      ✓ Save the tape copy created from the recovery logs.
      ✓ From the end of the log print output, save the message pair DLZ424I / DLZ425I.

      Below is an example of a DLZ424I / DLZ425I message pair:

DLZ424I	NO LOG TERMINATION RECORD FOUND FOR SCHEDULING RECORD
DLZ425I	PSBNAME = STBICL5P, SEQUENCE NUMBER = 00000264 DMBNAMEs = STDCDBPD, STDIDBPD, STDCY1PD, STDCX1PD, STDCX2PD, STDIX1PD

      The message pair shows the name of the DL/I PSB that was in use at the time of the DL/I batch abnormal termination, together with the names of the databases – if any – the application has changed since program start or the last DL/I checkpoint.

      Note: If – to run forward recovery at step 2. – all backups are to be used with the same set of log files (batch and online) – that is, all backups had been taken at around the same time – it's sufficient to create one tape-copy log only. However, if backups correlate to different sets of log

files, individual tape-copy logs have to be created through steps 1b. to step 1c. for recovery of the individual databases. In this case, different log print executions at step 1c. should still show the same DLZ424I / DLZ425I message pair.

2. DL/I forward recovery

   The DLZ424I / DLZ425I message pair in the log print output(s) obtained at step 1c. shows all databases ('DMBNAMEs = ...') with inflight changes at the time of the DL/I abnormal termination. These databases - if any - constitute a – **initial** – list of databases to be recovered.

   Note:

   a. If the DL/I abnormal termination happened because of a DLZ004I / DLZ005I error, and the database of the DLZ004I / DLZ005I data set is not on the **initial** list, it has to be **added**.

   b. Since in this recovery scenario it could be the case that before starting recovery, DL/I was executed again after the DL/I abnormal termination:

      When recovering a database, all related databases have to be recovered as well. This set of databases belonging together will then include

      - The main database (ESDS).
      - All index databases (primary and secondary indexes, KSDS) of the main database.
      - All databases logically related to the main database and their index databases, and any databases logically related to them and their index databases, etc.

      Of course – as an example – if the DL/I abnormal termination affected a HDAM database without indexes and logical relationship, the set contains one database only.

      By adding a DLZ004I / DLZ005I error database (only if applicable) and related databases, the mentioned initial list of databases to be recovered may become **extended**.

   c. Databases of the – now possibly **extended** – list of databases to be recovered, which have already been recovered for a DLZ004I / DLZ005I situation by “DL/I Batch Recovery Process VI” on page 86 above, have to be **removed** from the list. This gives the – **final** – list of databases to be recovered.

   d. You have to recover the full database. That is, if a database is defined on multiple data sets, you have to forward-recover each data set.

   For the individual dataset(s) of the **final** list of database(s) to be recovered, run the DL/I data set recovery utility using this input:

   - Backup (DL/I, VSAM, or other) of the data set.
   - DL/I change accumulation file (if available) and → tape-copy log file created at step 1c.

3. Backout inflight changes

   Having recovered the database(s) to the point of the (DLZ004I / DLZ005I) DL/I abnormal termination, uncommitted work may be existing that has now to be removed. You do this by running the DL/I data base backout utility on those databases, for which inflight changes were pending at the time of the DL/I abnormal termination, and that had not already been recovered for a DLZ004I / DLZ005I situation in “DL/I Batch Recovery Process VI” on page 86. To identify these databases, proceed as follows:

   - For the inflight PSB shown by the DLZ424I / DLZ425I message pair in the log print output(s) obtained at step 1c. determine, if there are database names shown in message DLZ425I via 'DMBNAMEs = ...' that coincide with a database name from the list of forward-recovered databases. For each match, this database has to be backed out for this PSB.

     Note: If no match is found for the PSB, or the PSB doesn't show any database(s) with inflight changes, backout can / need not be executed for this PSB.

   To remove inflight changes from the forward-recovered databases, run the DL/I data base backout utility on the PSB that was active at the time of the DL/I abnormal termination using the following input:

- Forward-recovered and unchanged databases that are accessed by the PSB, for which backout is executed.
- Tape-copy log file created at step 1c.

  Note: If more than one tape-copy log file had to be created, you can now use any one of them.
- One new LS input control statement for each forward-recovered database, which has been found as to have inflight changes with the PSB, for which backout is executed:

Column 1-2:	LS ... to identify the input control statement.
Column 4-10:	Name of the database to be backed out.

  A detailed description of the new LS input control statement can be found in "Backout per Database in Batch" on page 6.

Note:

- It's mandatory that you supply LS input control statements for the individual databases to be backed out. Otherwise, non-forward-recovered databases that may have been excluded at step 2c. might inadvertently be backed out as well and get corrupted.
- You don't need to create an output log since new backups should be taken after the backout and a new log cycle started (see below).
- Before invoking the backout utility, you have to position the input log tape behind the log file (standard labeled). If the log file is the first file on the tape you may use this command:

```
// MTC FSF,cuu,2
```

  cuu ... address of the tape device

This completes DL/I Batch Recovery Process VII.

You may now want to repeat all work for the recovered databases originating from the data that had to be backed out.

You should also consider to "Start a new Log Cycle and Take new Backups" on page 91, as described below, then you can resume DL/I batch or CICS/DLI online operation.

Start a new Log Cycle and Take new Backups

When one of the DL/ Batch Recovery Processes I, II, III, V, VI, or VII described above was run, the log data that had been written to the CICS journal resp. batch log file after the DLZ004I / DLZ005I error are not usable for a potential next DLZ004I / DLZ005I recovery of the same database(s). Because of that, a new log cycle has to be started. Before resuming CICS/DLI online or DL/I batch operation, you should therefore

- Reformat the CICS journals to mark the start of a new log cycle, and
- Create backups of all your databases.

The newly started log file(s) together with the new backups can then be used as base for any new DL/I recovery that should become needed.

Restart CICS and DL/I after DL/I Batch Recovery

Assuming CICS and DL/I had terminated normally before beginning, and now having completed "DL/I Batch Recovery Process I" on page 74, or "DL/I Batch Recovery Process II" on page 76 and, if required, "DL/I Batch Recovery Process III" on page 79, reformatted the CICS journals and taken new backups, you can restart CICS and DL/I either using

- START=COLD, or
- START=AUTO

and replying → **CONTINUE**, if you see messages

DLZ141I	I/O ERROR HAS OCCURRED IN DATABASE dbdname
DLZ142A	ENTER CONTINUE (ONLY IF DATABASE RECOVERED), IGNORE OR CANCEL

See "Responses to Message DLZ142A" on page 92 for a detailed description of the different responses that you can enter.

Responses to Message DLZ142A

These are the different responses you may enter to message DLZ142A and the effect they have on CICS and DL/I initialization:

CONTINUE

CICS Emergency Restart
Invokes DL/I dynamic backout, which resets all DL/I database changes inflight from the preceding CICS and DL/I abnormal termination. If – before the CICS Emergency Restart – you have forward-recovered the data set shown with message DLZ004I / DLZ005I of the database shown with message DLZ141I, this completes recovery for this database and ensures database integrity.

If you had not run forward recovery before the restart, entering 'CONTINUE' may corrupt the database.

CICS XRF takeover
Invokes DL/I dynamic backout in the same way as normal CICS Emergency Restart shown above. However, since at the time of the XRF takeover, it is impossible that the DLZ004I / DLZ005I data set has already been forward-recovered, entering 'CONTINUE' may corrupt the database.

CICS warm start
Continues CICS and DL/I initialization without additional recovery actions. If – before the restart – you have recovered the DLZ004I / DLZ005I database shown with message DLZ141I by an appropriate recovery process, database integrity is ensured.

If you had not run an appropriate recovery process, the DLZ004I / DLZ005I database may be corrupted.

IGNORE

CICS Emergency Restart, XRF takeover, CICS warm start
Leaves the database shown with message DLZ141I in closed state and you cannot access it or a database related to it when CICS and DL/I initialization has completed.

CICS Emergency Restart, XRF takeover
No backout is performed for this database or any other database, which is included in an online PSB (shown by message DFHER5723) that also references this database or one that is related to it.

When you see CICS message DFHER5723, enter 'GO', if you want CICS and DL/I initialization to continue.

Replying 'IGNORE' might be useful, if you have not yet (forward-) recovered the database shown with message DLZ141I, but want CICS and DL/I to be up and running. You can then perform DL/I recovery at a later point of time as described in "Recovering from missing or inappropriate DL/I Recovery" on page 72. But be aware that deferred recovery may be more time-consuming and roll back parts of the DL/I work you may have done in new CICS and DL/I cycles.

CANCEL

Terminates CICS and DL/I initialization
Use this option – for example, – when messages DLZ141I /DLZ142A reminded you that DL/I recovery has not yet been executed. You could run it now.

Note: Subsequent CICS and DL/I initializations will continue to show messages DLZ141I / DLZ142A until you reply 'CONTINUE' to message DLZ142A or perform a CICS COLD start. Be aware that both options may corrupt your (DLZ141I) database(s), when you haven't run the required DL/I recovery actions before.

Chapter 10. Messages and Codes

New and Changed Messages

The following messages have been changed for PH29950:

DLZ004I
DLZ005I
DLZ141I
DLZ142A
DLZ425I
DLZ428I
DLZ429I

The following messages are new for PH29950:

DLZ460I
DLZ461I
DLZ462I
DLZ464I
DLZ465I
DLZ885I
DLZ886I
DLZ887I

The following messages are new for DL/I VSE 1.12.1:

DBD810
DBD811
DBD812
DGEN205
DGEN258
DLZ029I
DLZ833I

The following messages have been changed for DL/I VSE 1.12.1:

DLZ035I
DLZ142A
DLZ252I

The following messages are new for DL/I VSE 1.12.0:

DLZ1064
DLZ1065
DLZ1066
DLZ150I
DLZ151I
DLZ152I
DLZ153I
DLZ154I
DLZ155I
DLZ156I
DLZ252I

The following messages have been changed for DL/I VSE 1.12.0:

DLZ1062
DLZ017I
DLZ144I

The following messages have been included earlier because they were new for DL/I VSE 1.11:

DLZ1028
DLZ1061
DLZ1062
DLZ019I
DLZ093I
DLZ094I
DLZ143I
DLZ144I
DLZ145I
DLZ146I
DLZ147I
DLZ250I
DLZ251I
DLZ431I

The following messages have been included earlier because they were new for DL/I DOS/VS 1.10:

DLZ1059
DLZ1060
DLZ109I
DLZ379I

The following messages have been included earlier because of changes for DL/I DOS/VS 1.10:

DLZ1055
DLZ002I
DLZ015I
DLZ037I
DLZ058I
DLZ096I

All other messages have been included because they were new for DL/I DOS/VS 1.9.

For messages not listed in the following section, refer to DL/ Messages and Codes.

DBD810 NPARTS PARAMETER SUPPORTED ONLY FOR HD, HDAM AND HIDAM DATA ORGANIZATION.

Explanation:
The NPARTS parameter can only be specified if the organization is HD, HDAM, or HIDAM.

User response:
Redefine either with ACCESS=HD | HDAM | HIDAM or without the NPARTS parameter.

DBD811 NPARTS NUMBER OF PARTITIONS SUB-OPERAND OMITTED OR INVALID.

Explanation:
The first suboperand of the NPARTS parameter indicating the number of partitions must be numeric and in the range of two to 128.

User response:
Specify a valid value.

DBD812 NPARTS PARTITION SELECTION ROUTINE NAME SUB-OPERAND OMITTED OR INVALID.

Explanation:
The second suboperand of the NPARTS parameter indicating the name of the partition selection routine must be 1 - 8 characters in length with the first character alphabetic.

User response:
Specify a valid routine name.

DGEN205 **MAXIMUM OF 254 DATASETS EXCEEDED.**

Explanation:
In the database Description (DBD) for a partitioned database, the number of data sets defined via DATASET statements multiplied by the number of partitions defined via the NPARTS parameter exceeds 254, which is the maximum allowed number of data sets in a database.

User response:
Either reduce the number of DATASET statements or the number of partitions defined via the NPARTS parameter.

DGEN258 **SEGMENT '*segmname*' ROUNDED SIZE (*n*) EXCEEDS MAXIMUM ALLOWED SEGMENT SIZE LESS ROUNDED ROOT ANCHOR POINT AND FSE SPACE (*m—p*) IN PARTITIONED DATABASE.**

Explanation:
In a partitioned database, segment lengths as well as the allocation of DL/I and VSAM-related information are rounded as described in "Data set utilization" on page 3. In this case, the rounded combined length of the named segment and its prefix exceeds the maximum allowed segment size, which includes rounded space for DL/I and VSAM-related information.

User response:
Shorten the segment or use a higher VSAM block size.

DLZ1028 **INVALID BUFFER SPECIFICATION IN HDBFR OPERAND, IT WAS SET TO *nn*.**

Explanation

The number of buffers specified in the HDBFR operand of the DLZACT TYPE=BUFFER macro instruction was not a numeric value in the range 2 - 255. It is set to the number indicated in the message text.

DLZ1055 **REMOTE PSB WITH LOCAL COMPONENT NOT ALLOWED IN DFHMIRS TYPE=PROGRAM STATEMENT, ENTRY IGNORED IN DFHMIRS TYPE=PROGRAM STATEMENT**

Explanation

Program DFHMIRS, the CICS intercommunication mirror program, is not allowed to schedule an RPSB with a local component. The name of this RPSB was specified in the TYPE=PROGRAM statement for DFHMIRS. Therefore, the RPSB entry in the DFHMIRS TYPE=PROGRAM statement will be ignored. Definition of this PSB as a remote PSB is allowed.

DLZ1059 **HSMODE = *operand* INVALID, HSMODE = BELOW ASSUMED**

Explanation

The HSMODE operand of the DLZACT TYPE=CONFIG statement is incorrectly specified. The allowed HSMODE operands are ANY or BELOW.

The default HSMODE=BELOW will be assumed.

DLZ1060 **'DFHMIR' TYPE=PROGRAM STATEMENT CHANGED INTO 'DFHMIRS' TYPE=PROGRAM STATEMENT**

Explanation

With CICS/VSE 2.2, the CICS intercommunication mirror program name has been changed from DFHMIR to DFHMIRS. However, DL/I still accepts the old name DFHMIR in the TYPE=PROGRAM statement, but creates an entry for DFHMIRS in the Application Control Table (ACT) phase.

To avoid the MNOTE in the ACT generation, specify DFHMIRS in the TYPE=PROGRAM statement of the ACT source.

DLZ1061 **PSBLOC=*operand* INVALID. PSBLOC=BELOW ASSUMED**

Explanation

The default value of PSBLOC=BELOW was assumed for the PSBLOC parameter in the DLZACT TYPE=CONFIG statement. PSBs will be loaded into storage below the 16 MB line during CICS DL/I initialization.

DLZ1062 **SVA PARAMETER HAS BEEN DISABLED. DL/I PHASES RESIDING IN THE SVA ARE ACCEPTED**

Explanation

The SVA parameter could be used until z/VSE 4.2 for DL/I VSE 1.11 running with CICS TS in a coexistence environment together with DL/I DOS/VS 1.10 and CICS/VSE 2.3 to control usage of SVA resident DL/I action modules. As the coexistence environment is no more supported since z/VSE 4.3, the SVA parameter has been disabled. When DL/I phases have been loaded into the SVA, they are accepted and will be used from there.

DLZ1064 **USREXIT = *operand* INVALID, USREXIT=AMODE-24 ASSUMED**

Explanation

An invalid operand has been specified for parameter USREXIT in the DLZACT TYPE=CONFIG statement. DL/I user exit routines will be called in the addressing mode of their AMODE library attribute. PSBs and HD buffers will be allocated below the 16 MB line.

DLZ1065 **HDMODE = *operand* INVALID, HDMODE=BELOW ASSUMED**

Explanation

An invalid operand has been specified for parameter HDMODE in the DLZACT TYPE=CONFIG statement. HD buffers will be allocated below the 16 MB line.

DLZ1066 **parameter = ANY IS CONFLICTING WITH USREXIT=AMODE-24, parameter = BELOW ASSUMED**

Explanation

Either PSBLOC=ANY or HDMODE=ANY has been specified to allocate PSBs or HD buffers above the 16 MB line. This is not supported, when there are DL/I user exit routines that need to run in AMODE 24.

DLZ002I **BATCH DL/I ABNORMAL TERMINATION COMPLETE**

Explanation

This message follows DLZ001I if the buffer pool records were successfully purged (written) and the databases closed. A storage dump is produced.

The following provides information on the storage dump and the layout of the save areas.

Storage Dump – The following should be noted from the storage dump:

Register	Contents
2	Addressable part of SCD (address of SCD plus 96 bytes).
3	Address of user save area if abnormal end was entered with a pseudo-abend.If entered for a program check or abnormal end, register 3 value is not set.
4	Contains original 2–byte error code at entry to pseudo-abend.
5	Address of STXIT AB save area if abend indicator shows that STXIT AB processing has been entered.
6	Address of the application program entry point.
7	Address of STXIT PC or pseudo–abend save area.
8	Address of PST.

Changes to Save Area Layout

As of DL/I 1.10, the save area has been extended to the new layout that is introduced with VSE/ESA 1.3. Since DL/I 1.10:

- If STXIT PC or STXIT AB has been issued from within DL/I, the new layout will be used.
- If STXIT PC has been set from a PL/I application, the old layout is still valid.

Save Area Layout for STXIT PC

At label SCDABSAV in the SCD is a pointer to the STXIT or pseudo-abend save area which contains the following information:

```
   Dec   (Hex)  Length  Description

   -32  (-20)      32  Constant: *****
DL/I ABEND SAVE AREA *****
     0   (00)       8  Program old PSW
(program check only; BC mode PSW)
     8   (08)      64  Registers 0 to 15
    72   (48)       8  EC mode PSW
    80   (50)       8  Reserved
    88   (58)      64  AB exit: Error
information dependent on cancel code
   152   (98)      64  Save area for
access registers 0 to 15
                       (not applicable for
DL/I)
   216   (D8)      17  Constant: ABEND
INDICATORS*
   233   (E9)       1  Abend reason
indicator:
                         X'80' - Entered
by STXIT AB'
                         X'40' - Entered
by STXIT PC'
                         X'20' - Abend in
progress'
                         X'10' - Buffer
pool or data base damage'
                         X'08' - Buffer
pool unload and close complete'
   234   (EA)       1  STXIT AB reason
code as returned by the system
                       (low order byte of
register 0)
   235   (EB)      25  Constant:
*PSTFNCNT,RTCDE,OFFST-XXH
   260  (104)       1  PSTFNCTN (function
code)
   261  (105)       1  PSTRTCDE (return
code)
   262  (106)       2  PSTOFFST (offset)
```

```
   264  (108)      24  Constant:
PSTBLKNM,BYTNM,DATA-FFF-
   288  (120)       4  PSTBLKNM (relative
block number)
   292  (124)       4  PSTBYTNM (RBA or
relative record number)
   296  (128)       4  PSTDATA (address of
request data)
   300  (126)      24  Constant: PSTSV1
THROUGH PSTV3---
   324  (144)      72  PSTV1 (save area)
   396  (186)      72  PSTV2 (save area)
   468  (1D4)      72  PSTV3 (save area)
```

Save Area Layout for STXIT AB

If the abend indicator shows that STXIT AB processing has been entered, the STXIT AB save area can be located as follows:

1. Find the address of the SCD extension (label SCDEXTBA).
2. Find the address of the STXIT AB save area (label SCDEABSV).

This address points to the STXIT AB save area which contains the following information:

```
   Dec   (Hex)  Length  Description

     0   (00)       8  Program old PSW (BC
mode PSW)
     8   (08)      64  Registers 0 to 15
    72   (48)       8  EC mode PSW
    80   (50)     136  Information not
applicable for DL/I STXIT PC
```

The entry point address of the application program is in the SCDAPSTR field of the SCD. The DL/I high address rounded up to page boundary is in the field SCDDLIUP.

User response

Take appropriate action.

Note: For information on additional aids for interpreting and debugging dumps, refer to the *DL/I Diagnostic Guide* under "Chapter: Interpreting And Debugging Dumps".

DLZ004I **ERROR ACCESSING FILE *file name*, VSAM CODE=*xxyy***

Explanation

An I/O error occurred during an attempt to read or write to the file *file name*.

For an explanation of the VSAM return code *xx* and the VSAM error code *yy* refer to z/VSE Messages and Codes, Volume 2 , Section "VSE/VSAM Return and Error Codes".

This message can occur on an I/O error either when accessing a data base data set or when writing to a disk log file.

User response

Remove the error condition, perform data base recovery, and resubmit the job. For further information refer to Chapter 9, “Recovery after a DLZ004I or DLZ005I Error,” on page 69.

DLZ005I ***file name* STOPPED, OUT OF SPACE, VSAM CODE=*xxyy***

Explanation

An attempt was made to add a record to the file *file name* and there was no more EXTENT allocation.

For an explanation of the VSAM return code *xx* and the VSAM error code *yy* refer to z/VSE Messages and Codes, Volume 2 , Section "VSE/VSAM Return and Error Codes".

User response

Provide more disk space, perform data base recovery, and resubmit the job. For further information, refer to Chapter 9, “Recovery after a DLZ004I or DLZ005I Error,” on page 69.

DLZ015I **PARAMETER STATEMENT DATA INVALID**

Explanation

A DL/I parameter statement contains one of the following error conditions:

- Data did not start in column 1.
- A field length was invalid.
- A required field was omitted.
- An expected continuation statement was not found.
- A continuation statement was specified for SYSLOG input.
- The LOG parameter was specified a second time.
- The TRACE parameter was specified a second time.
- The HSMODE parameter was not specified as HSMODE=ANY or HSMODE=BELOW.
- ULR used without DLZURGU0.
- PLU used without DLZURGU0.
- PLU used and the specified PSB references a DBD or an index DBD.

User response

Correct the error and execute the job again.

DLZ017I **PSB VERSION INVALID, DL/I TERMINATED**

Explanation

While loading the requested PSB, initialization detected that the block was created with an incompatible version of DL/I.

User response

Rebuild the PSB using the application control block utility DLZUACB0 provided with the current level of DL/I.

DLZ019I **BUFFER NUMBER IN {HDBFR| HSBFR} STATEMENT TOO {LOW| HIGH}, SET TO {MINIMUM| DEFAULT}**

Explanation

One of the buffer number specifications did not fulfill the minimum requirements, or the specification was too high. DL/I resets the values to the minimum (if it was too low) or the default. See *DL/I DOS/VS Resource Definition and Utilities* for information on the requirements.

User response

None.

DLZ029I **PARTITION SELECTION MODULE *modname* FOR DBD *dbdname* NOT FOUND**

Explanation:
Partition selection module *modname* is required by the Database Description (DBD) *dbdname*. Module *modname* was not found in any of the libraries defined in the search chain and, therefore, was not loaded.

User response

If message DLZ040A has not been stopped by a previous RUN response, it is issued after this message.

Respond to DLZ040A as follows:

- To continue with DL/I and CICS initialization, enter either:

  GO
  This database will be disabled.

  RUN
  This database will be disabled. (DL/I will stop issuing DLZ040A messages.)

- To stop DL/I and CICS initialization, enter either:

  CANCEL
  To end DL/I and CICS initialization.

  DUMP
  To end DL/I and CICS initialization and DUMP the contents of storage at this point.

To correct this problem, catalog and link-edit the partition selection module as a phase in a library using the system link-edit procedure and ensure that this library is defined in the search chain. Rerun the CICS and DL/I initialization job.

DLZ035I **INVALID DATABASE ORGANIZATION FOR PARTIAL DB LOAD OR PARTIAL DB REORG**

Explanation

- PLS/PLC used in the parameter statement and the specified PSB references a DBD with HSAM/ SHSAM or HISAM/SHISAM organization or HDAM/ HIDAM organization defined with multiple DATASET statements or partitioning.
- Partial database reorganization with HDAM/HIDAM organization defined with multiple DATASET statements or partitioning.

User response

- Load the database with one step (DLI in parameter statement).
- Or run partial database reorganization with HDAM/ HIDAM organizations only, if the database is not defined with multiple DATASET statements or partitioning.

DLZ037I **DL/I STATUS=xx RETURNED, STMT 'yyyyyyyy', PROGRAM 'zzzzzzzz' TERMINATED**

Explanation

A DL/I status code (xx) indicating an abnormal condition was returned to the application program (zzzzzzzz) and the program was abnormally terminated. See *DL/I Messages and Codes* under "DL/I Status Codes" for an explanation of the status codes. The <yyyyyyyy> is the statement number of the command that generated the status code, if available. For PL/I, the statement number will be taken from columns 73 to 80 of the listing produced by the CICS translator. For COBOL, the statement number will be six characters taken from column 1 to 6.

User response

Correct the error and rerun the application program.

DLZ050I **DEVICE MISMATCH FOUND IN DBD *dbdname***

Explanation

DEVICE=CKD was specified in DATASET statement of the database definition, but FBA was found in the device characteristics extracted during open of the data set. For HD/HDAM/HIDAM databases the value of the SCAN parameter is set to the default of FBA devices (32768). Processing continues and the return code is set to 4. For HSAM/SHSAM databases the mismatch leads to a termination. The return code is set to 12.

User response

Correct the DEVICE parameter in the DATASET statement of DBDGEN for HSAM/SHSAM databases and rerun the program. For HD/HDAM/HIDAM databases the DEVICE parameter should be corrected if the SCAN parameter should not have the default value.

DLZ051I **GETVIS ERROR, RETURNCODE = *rc***

Explanation

The OPEN/CLOSE routine encountered an error when executing a GETVCE request. The program terminated. The return code is given in decimal format. For an explanation of the GETVCE return code *rc*, refer to *z/VSE Messages and Codes*, in the section that describes "VSE/Advanced Functions Codes and SVC Errors".

User response

Correct the error and rerun the program.

DLZ053I **DL/I INITIALIZATION COMPLETE [FOR XRF STANDBY | FOR XRF TAKEOVER]**

Explanation

DL/I online initialization was successful and no errors were detected.

The message "DLZ053I DL/I INITIALIZATION COMPLETE" is issued within an active CICS XRF environment or in a CICS system without XRF.

The message "DLZ053I...FOR XRF STANDBY" and "DLZ053I...FOR XRF TAKEOVER" is issued in an alternate CICS XRF environment.

The DL/I initialization in the alternate CICS XRF environment is a two-stage process. 'STANDBY' indicates that the alternate DL/I initialization has completed but without open of the databases. DL/I is waiting in a standby mode and is ready to take over if the active DL/I CICS system terminates due to an error situation or a console command. 'TAKEOVER' indicates that the alternate DL/I initialization has completed including the open of the databases and takeover has finished.

User response

None required.

DLZ054I **DL/I INITIALIZATION ERROR(S) DETECTED [FOR XRF STANDBY | FOR XRF TAKEOVER]**

Explanation

DL/I online initialization was successful, but one or more errors were detected.

The message "DLZ054I DL/I INITIALIZATION ERROR(S) DETECTED" is issued within an active CICS XRF environment or in a CICS system without XRF.

The message "DLZ054I...FOR XRF STANDBY" and "DLZ054I...FOR XRF TAKEOVER" is issued in an alternate CICS XRF environment.

The DL/I initialization in the alternate XRF environment is a two-stage process. 'STANDBY' indicates that the alternate DL/I initialization has completed but without open of the databases. DL/I is waiting in a standby mode and is ready to take over if the active DL/I CICS system terminates due to an error situation or a console command. 'TAKEOVER' indicates that the alternate DL/I initialization has completed including the open of the databases and takeover has finished. Note that the error indication from first stage initialization is propagated.

User response

None required.

DLZ058I **INSUFFICIENT STORAGE TO START DL/I**

Explanation

The virtual storage area is too small to hold the DL/I modules or buffer pools.

User response

For Batch Environment

This message is followed by an immediate abnormal termination and dump.

Check the following:

- The HDBFR parameter in the DL/I parameter statement can be used to decrease the number of buffers required. It is possible that the parameter is not read because of a missing continuation character in column 72.

  If this caused the problem, correct the DL/I parameter statement and run the job again, or proceed as shown below under *Corrective Action*.
- The eligible DL/I action modules have not been loaded in the shared virtual area (SVA). For a complete list of SVA-eligible DL/I phases, refer to "SVA Loading for DL/I" on page 19.

  If this caused the problem, install the DL/I action modules in the SVA and run the job again. Alternatively, you can proceed as shown under *Corrective Action*.
- There is insufficient program space in the program area to load the application. If this is the case, proceed as shown under *Corrective Action*.

Corrective Action:

- Increase the SIZE parameter in the EXEC statement and resubmit the job,

  or
- Increase the partition size and the SIZE parameter in the EXEC statement and resubmit the job.

For Online Environment

If message DLZ040A has not been stopped by a previous RUN response, it is issued after this message.

Respond to DLZ040A as follows:

- To continue only with CICS initialization, enter either GO or RUN.

  DL/I will not be initialized. A dummy DL/I nucleus will be used which returns control to the application program when DL/I is called without performing any DL/I functions.
- To stop DL/I and CICS initialization, enter either:

  CANCEL to end DL/I and CICS initialization.
  DUMP to end DL/I and CICS initialization, and to dump the contents of storage at this point.

To correct this problem, check that:

- The HDBFR parameter in the DLZACT TYPE=BUFFER statement can be used to decrease the number of buffers required. It is possible that the parameter has no effect because of an error in the DLZACT statements that were used to build this nucleus.

  If this caused the problem, correct the DLZACT TYPE=BUFFER statements and rebuild the DL/I nucleus (ACT) by using the ACTGEN procedure, or proceed as shown below under *Corrective Action*.
- The eligible DL/I action modules have not been loaded in the shared virtual area (SVA). (For a complete list of SVA-eligible DL/I phases, refer to "SVA Loading for DL/I" on page 19.)

  If this caused the problem, install the DL/I action modules in the SVA, or proceed as shown below under *Corrective Action*.

Corrective Action:

- Increase the SIZE parameter in the EXEC statement and rerun the CICS/DLI initialization job,

  or
- Increase the partition size and the SIZE parameter in the EXEC statement and resubmit the job.

DLZ093I **MPS STARTED WITH APPLID=*xxxxxxxx***

Explanation

The Master Partition Controller (MPC) for DL/I Multiple Partition Support (MPS) started successfully. Batch jobs for MPS can now be submitted. *xxxxxxxx* shows the CICS (generic) applid of the CICS/DLI partition.

User response

None required.

DLZ094I **MPS WITH APPLID=*xxxxxxxx* STOPPED {ABNORMALLY|NORMALLY}**

Explanation

Multiple Partition support (MPS) ended abnormally or normally. If it ended abnormally the previous message gives the reason. No further batch jobs can be run in this CICS/MPS environment. *xxxxxxxx* shows the CICS (generic) applid of the CICS/DLI partition.

User response

None required.

DLZ096I **STXIT {AB|PC} ENTERED, MPS BATCH PARTITION ENDED ABNORMALLY**

Explanation

Control was passed to the STXIT AB or STXIT PC entry point in the MPS batch abend handler routine (DLZMABND).

User response

Examine the log and dump (if printed), to determine the cause of the error. The corresponding register save areas are immediately preceded in the dump of module DLZMPI00 by the eye-catchers 'AB SAVE' or 'PC SAVE' respectively. The 1-byte reason code for the AB STXIT entry is preceded by 'AB REASON CODE'. In the dump, Register 3 contains the address of the applicable save area.

Note: As of DL/I 1.10, the save area has been extended to the new layout that is introduced with VSE/ESA 1.3. Since DL/I 1.10 the following applies:

- If STXIT PC or STXIT AB has been issued from within DL/I, the new layout will be used.
- If STXIT PC has been set from a PL/I application, the old layout is still valid.

For information on additional aids for interpreting and debugging dumps, refer to *DL/I Logic Extensions* under "Diagnostic Aids for MPS Messages".

DLZ107I **RECORD LENGTH (*nnnnn*) EXCEEDS DEVICE LIMIT**

Explanation

The specified record size for HSAM/SHSAM databases exceeds the maximum record length:

```
track capacity for CKD devices,
32766 bytes    for FBA devices.
```

User response

Shorten the segment or split it into one or more smaller segments with a parent-child relation, or change the device type, or correct your record length specification.

DLZ140I **CICS JOURNAL NOT ACTIVE BUT REQUIRED FOR XRF SUPPORT**

Explanation

A DL/I CICS XRF environment requires CICS journaling for DL/I logging. This message is issued if you run CICS without journaling or if CICS journaling was suppressed by UPSI bits 6 and 7 in the CICS initialization job. The message DLZ140I may be followed by message DLZ040A.

User response

If message DLZ040A has not been stopped by a previous RUN response, it is issued after this message. Enter CANCEL as response to message DLZ140I to terminate the DL/I and CICS initialization. In order to enter other possible replies, refer to message DLZ049I. In order to correct the problem, set UPSI bits 6 and 7 in the CICS initialization job to 0 to activate CICS journaling. Rerun the job.

DLZ141I **I/O ERROR HAS OCCURRED IN DATABASE *dbdname***

Explanation

During CICS warm start, emergency restart or takeover the DL/I recovery routines found a database stop record in the CICS restart data set indicating that an I/O error had occurred for the database *dbdname*. Message DLZ004I or DLZ005I was previously issued. Message DLZ141I is followed by DLZ142A. Recovery actions have to be taken to physically repair the database.

For more information, refer to the message DLZ142A.

User response

None required.

DLZ142A **ENTER CONTINUE (ONLY IF DATABASE RECOVERED), IGNORE OR CANCEL**

Explanation

Message DLZ142A follows DLZ141I.

User response

Enter 'CONTINUE', 'IGNORE', or 'CANCEL' as described in “Responses to Message DLZ142A” on page 92.

DLZ143I **MPS BATCH CONNECT REQUEST FOR {PARTID=*yy*| APPLID= *xxxxxxxx*| PARTID=*yy*,APPLID=*xxxxxxxx*}**

Explanation

An MPS batch job has been started. DL/I calls will be executed in the CICS/DLI MPS system as shown in the PARTID/APPLID parameter of the message text. *yy* shows the VSE partition id, *xxxxxxxx* the CICS (generic) applid of the CICS/DLI partition.

User response

None required.

DLZ144I **CONTROL BLOCK DLZMPS INVALID, WILL BE REBUILT [AT NEXT MPC START]**

Explanation

The beginning of the MPS control block does not contain the expected eye catcher and is therefore considered to be invalid. This is an internal error.

User response

If the message was shown with the CSDA transaction at starting the MPS Master Partition Controller (MPC), for example, during CICS/DLI initialization, the MPS control block will be directly rebuilt and MPC initialization continues. MPS batch jobs connecting to this CICS/DLI partition can be run.

If the message was shown with the CSDD MPC stop transaction, for example, during CICS/DLI termination, this can be treated as only informational for the partition, where CSDD had been entered. MPC operation in this partition can be started again through the CSDA start transaction, for example, during the next CICS/DLI initialization.

In both cases, MPS batch connections to any other CICS/DLI partition, where MPC operation had been active before message DLZ144I was issued, will fail. In these partitions MPC needs first to be stopped through the CSDD transaction and then started again through the CSDA transaction.

DLZ145I MORE THAN ONE MPS ACTIVE, SPECIFY 'PARTID=' AND/OR 'APPLID='

Explanation

An MPS batch job without a PARTID/APPLID parameter has been submitted, but more than one CICS/DLI MPS system is active. MPS cannot determine, to which of these systems it should connect.

User response

Specify an according PARTID and/or APPLID parameter in the DLI/DLR statement of the MPS batch job. Else ensure that only one CICS/DLI MPS system is active.

DLZ146I RESTART PARTID/APPLID COULD NOT BE VERIFIED

Explanation

An MPS batch job has been restarted. The restarted MPS batch job expects the same CICS/DLI online environment as at the time of its abend. Therefore, it will try to connect to the same partition and same applid. Either no CICS/DLI MPS system is active in this partition, or the CICS applid has changed. Message DLZ147A will follow.

User response

See message DLZ147A.

DLZ147A ENTER 'CANCEL' TO END, OR NEW 'PARTID=' AND 'APPLID=' PARAMETER

Explanation

Message DLZ146I has been written before. The job can either be cancelled or should be executed in a different CICS/DLI MPS environment.

User response

Enter 'CANCEL', if you do not want to correct the error now. The job can then be restarted later. If execution should continue, a CICS/DLI MPS system must be active with the same DL/I MPS characteristics as at the time of the last execution. This includes the same Temporary Storage Queue (TSQ) and all required databases. When the correct CICS/DLI MPS system has been determined, enter:

```
PARTID=yy,APPLID=xxxxxxxx
```

yy denotes the VSE partition id, *xxxxxxxx* the CICS (generic) applid of the CICS/DLI MPS system, where the job should be restarted.

DLZ150I LOAD FOR DLZMPX00 FAILED, CDLOAD RETURN CODE = rc

Explanation

The language interface module DLZLX000 has tried to load the DL/I connector module DLZMPX00 on a DL/I call entered via AIBTDLI. The load has failed for the reason given in the return code 'rc' shown in decimal format. The program is canceled.

For an explanation of the CDLOAD return code *rc* refer to *z/VSE Messages and Codes*, Section "VSE/Advanced Functions Codes and SVC errors".

User response

Correct the error and rerun the program.

DLZ151I NON-COMPATIBLE ENVIRONMENT FOR AIBTDLI INTERFACE

Explanation

A user program has issued a DL/I call via AIBTDLI. The language interface module DLZLX000 has detected that either

- DLZRRC00 for DL/I batch
- DLZMPI00 for MPS batch or
- CICS/DLI online

is already active in this partition. DL/I calls via AIBTDLI are only supported, when no specific DL/I environment has been started in the partition. That means that DLZRRC00, DLZMPI00 or CICS/DLI online may not be running. The program is abnormally terminated.

User response

Rerun the program in a partition without a pre-established DL/I environment.

DLZ152I **DL/I EXIT ROUTINE DLZBSEOT NOT LOADED IN SVA**

Explanation

A user program has issued a DL/I call via AIBTDLI, but the DL/I task termination routine DLZBSEOT was not residing in the SVA. The program is canceled.

User response

Load DLZBSEOT into the SVA and rerun the program.

DLZ153I **GETVIS FOR AIB FAILED, RETURN CODE = rc**

Explanation

The DL/I connector module DLZMPX00 encountered an error while executing a GETVIS request for the Application Interface Block (AIB) on a DL/I call passed via AIBTDLI. The GETVIS return code 'rc' is given in decimal format. The program is canceled.

For an explanation of the GETVIS return code *rc* refer to *z/VSE Messages and Codes*, Section "VSE/Advanced Functions Codes and SVC errors".

User response

Correct the error and rerun the program.

DLZ154I **MISSING OR INVALID AIB PARAMETER AT SCHEDULING CALL**

Explanation

On a scheduling call entered via AIBTDLI, the mandatory parameter, where DL/I returns the address of the AIB, was missing or had an invalid address. The program is canceled.

User response

Correct the CALL statement and rerun the program.

DLZ155I **WRONG 'PARTID=' OR 'APPLID=' PARAMETER AT SCHEDULING CALL**

Explanation

The 'PARTID=' or 'APPLID=' parameter was incorrectly specified on a scheduling call entered via AIBTDLI.

User response

Correct the CALL statement and rerun the program.

DLZ156I **ERROR AT ONLINE PARTITION IN *id***

Explanation

The CICS/DLI online partition identified by *id* has encountered an error while processing a DL/I call passed via AIBTDLI.

User response

To locate the problem, check the console error messages for this partition, which were displayed before this message.

DLZ250I **CICS REGISTRATION CALL FAILED, DL/I TASK TERMINATED**

Explanation

A CICS transaction tried to register as a DL/I task, but the registration failed.

User response

This is an internal error, which inhibits CICS/DLI online operation. Report the problem to your IBM support center.

DLZ251I **DL/I NOT ACTIVE, TASK TERMINATED**

Explanation

A CICS transaction issued a DL/I call, but the DL/I online system is not active. Errors have been encountered during CICS/DLI online initialization.

User response

Correct the errors. Ensure that message DLZ053I is shown during CICS/DLI online initialization.

DLZ252I ***dli-rel*** **NOT SUPPORTED ON THIS VSE RELEASE**

Explanation

An attempt was made to run:

- DL/I DOS/VS 1.10 or DL/I VSE 1.11 on z/VSE 4.3 or later, or
- DL/I VSE 1.12.1 on z/VSE 6.1 or an earlier release of z/VSE

This has been rejected. Starting with z/VSE 4.3, DL/I VSE 1.12 is the only DL/I release supported. If DL/I VSE 1.12.1 is to be used, z/VSE 6.2 is needed.

User response:
If DL/I is to be used on this VSE system, it should be DL/I VSE 1.12.0. To run DL/I VSE 1.12.1, z/VSE 6.2 is needed.

DLZ379I **ERROR - INTERNAL WORKING STORAGE HAS BEEN EXCEEDED**

Explanation

The *change accumulation* or *recovery* utility has detected that one of the internal work areas has been exceeded. The program terminates and issues the message DLZ385I.

The size of these (GETVIS) work areas depends mainly on the CI size of the processed database. Also, consider that there may be other CUMIN or CUMOUT records to be processed that require work areas larger than the standard size.

The work areas and their standard sizes are as follows:

- *Recovery Utility* (DLZURDB0)

  CUMIN input work area size:

  CI size x 2.5, or
  63 K if MAXI is specified in S-card.

  The minimum is 10 K, the maximum is 63 K .
- *Change Accumulation Utility* (DLZUCUM0)
  - CI work area size (if CUMIN provided):

    CI size from first log record, or (if greater)
    nnK CI size specified in ID-card (nn = 04 ... 30), or
    30 K if MAXI is specified in ID-card.

    The minimum is 4 K .
  - CUMIN input work area size:

    (Previous) CI work area size x 2.5.

    The minimum is 10 K, the maximum is 63 K.
  - CUMOUT output work area size:

    (Previous) CI work area size x 2.5.

    The minimum is 10 K, the maximum is 63 K.

User response

In the applicable utility:

1. Specify MAXI in column 25-28 of the S- or ID-card (control statement).
2. Resubmit the job.

DLZ425I **PSBNAME =** ***psbname*****, SEQUENCE NUMBER =** ***nnnnnnnn*** **[DMBNAMEs =** ***dmbname1, dmbame2*****, ...]**

Explanation

The log print utility did not find a termination record to match the scheduling record whose PSB name is *psbname* and whose sequence number is *nnnnnnnn*. This message follows message DLZ424I.

In addition, when printing of log records was stopped at a defined sequence number that was specified as parameter on the LO control statement, the *dmbnames* of all databases (DMBs) that were changed – that is, for which DL/I data base log record(s) had been written – since the named PSB had been scheduled at the record with the indicated sequence number, or since the last DL/I checkpoint had been written for this PSB (batch execution only), are also shown.

For a description of the log sequence number parameter, refer to “Stop Log Print on Log Sequence Number” on page 5.

User response

None.

DLZ428I **ERROR ON FILENAME -** ***file name*****, LOG COPY HALTED**

Explanation:
The log print utility detected an invalid record on the file name indicated in the message text. The error condition is identified by a previous error message.

The copy condition is identified by a previous error message. The copy condition is identified by a previous error message. The copy function that was requested by specifying COPY or CPYR on the LO input control statement, is now stopped. Further processing on this file or any subsequent file is terminated.

User response:
None.

DLZ429I LOG COPY SUCCESSFUL

Explanation:
The log print utility successfully copied the contents of the input log file to a new log tape file. The copy function was requested by specifying COPY or CPYR on the LO input control statement. The new log tape file may be used as input to the backout utility.

User response:
None.

DLZ431I WARNING - RECORDS ON THE LOGFILE SEEM TO BE INCOMPLETE

Explanation

An open record on the input log file is missing. The log file or log file sequence may be incomplete.

User response

Ensure that the set of input logfile(s) is complete. Resubmit the job, if necessary.

DLZ460I ERROR – INVALID SEQUENCE NUMBER

Explanation

The log sequence number specified in the LO control statement is invalid for one of the following reasons:

- The number contained other than hexadecimal characters (0 - 9, A to F).
- The number was longer than 8 characters.
- The number did not begin in column 54.
- The column immediately following the number was not blank.

User response

Correct the sequence number and resubmit the job.

DLZ461I ERROR – INVALID SEQUENCE NUMBER VERSION

Explanation

The log sequence number version specified in the LO control statement is invalid for one of the following reasons:

- The version contained other than numeric characters.
- The version was longer than 2 characters.
- The version did not begin in column 63.
- The version was 0.
- The column immediately following the version was not blank.
- The version was specified without a sequence number.

User response

Correct the sequence number version and resubmit the job.

DLZ462I END SEQUENCE NUMBER EXCEEDED ON FILENAME – *file name*

Explanation

The log print utility encountered a DL/I log record whose sequence number / version is higher than the sequence number / version specified in the LO input control statement. Processing on this file and any subsequent file is terminated.

User response

None.

DLZ464I ERROR – A DUPLICATE DATA BASE NAME HAS BEEN ENCOUNTERED

Explanation

The same DBD name was specified on more than one LS statement.

User response

Correct the control statement, and resubmit the job.

DLZ465I ERROR - TOO MANY DATA BASES

Explanation

The number of data bases passed by LS statements exceeds 16.

User response

If you want to back out more than 16 databases for a PSB, you have to submit multiple jobs with a maximum of 16 different databases supplied by LS control statements in one job.

DLZ833I **INVALID ALIGNMENT OF SPACE OFFSET OR LENGTH, DL/I TERMINATED**

Explanation:
For a partitioned database, either the DL/I buffer handler detected that offset of or space size represented by a free space element was not rounded as required, or DL/I space management found that the offset of requested segment space was not rounded as required.

User response:
See "DL/I System/Task Abnormal Termination" of DL/ Messages and Codes.

DLZ885I **BACKOUT COMPLETE FOR DATA BASE *dbdname***

Explanation

The data base backout utility has successfully completed the requested data base backout for data base *dbdname*.

User response

None.

DLZ886I **NO DATA BASE RECORDS READ FOR DATA BASE *dbdname***

Explanation

The log file supplied contained no data base log records for database *dbdname* and the executed PSB. This message may also occur if the task(s) that used the executed PSB terminated normally.

User response

None.

DLZ887I **NO BACKOUT DONE FOR PSB *psbname***

Explanation

The log file supplied contained no data base log records for a database entered by an LI control statement and the PSB specified. This message may also occur if the task(s) that used the named PSB terminated normally.

User response

None.

New DL/I CICS Return Codes

Type of Call	UIBFCTR	UIBDLTR	Explanation
Scheduling	0C	09	Internal error at scheduling conflict; contact IBM support

Type of Call	AIBFCTR	AIBDLTR	Explanation
Scheduling	08	0A	Roll Back call when not scheduled
Any	FF	00	Unrecoverable error using AIBTDLI interface.

Appendix A. Sample Job Streams for DL/I 31-Bit Applications

This section contains sample job streams showing, how to create DL/I 31-bit applications for the different environments, using different programming interfaces and languages.

DL/I - Online Applications

For DL/I online applications written in COBOL for VSE/ESA the job control requirements are the same as for DL/I online applications written in COBOL II.

For DL/I online applications written in PL/I for VSE/ESA the formerly required inclusion of object module DFHPL1I must be replaced through the inclusion of DFHELII. The inclusion of object module PLISHRE must be dropped. All programs should be compiled with option SYSTEM(CICS).

DL/I online applications written in High Level Assembler for VSE can be assembled and link-edited as before.

COBOL Online Example

The following example illustrates the job control statements needed to compile and link-edit a DL/I online application written in COBOL for VSE/ESA or COBOL II. IGYCRCTL is the name for both compilers.

```
// JOB CBLSAMPL
// DLBL IJSYSPH,'COBOL TRANSLATION',yyyy/ddd
// EXTENT SYSPCH,extent information
ASSGN SYSPCH,DISK,VOL=volid,SHR
// LIBDEF *,SEARCH=search-libraries
// LIBDEF PHASE,CATALOG=catalog-library
// EXEC DFHECP1$,SIZE=...
 CBL XOPTS(CICS,DLI,COBOL2),LIB,RES,RENT,DATA(31)
        .
        .
   SOURCE DECK
        .
        .
        .
        .
/*
CLOSE SYSPCH,punch
// DLBL IJSYSIN,'COBOL TRANSLATION',yyyy/ddd
// EXTENT SYSIPT
ASSGN SYSIPT,DISK,VOL=volid,SHR
// OPTION NODECK,CATAL
   PHASE CBLSAMPL,*
   INCLUDE DFHELII
// EXEC IGYCRCTL,SIZE=...
// EXEC LNKEDT
/&
// JOB RESET
CLOSE SYSIPT,reader
/&
```

Figure 17. COBOL Online with HLPI or CALL Interface

Note: When the DL/I CALL interface is used the 'DLI' parameter in the CBL XOPTS statement should be omitted.

As an alternative to DFHELII, the interface DFHECI can be used.

PL/I for VSE/ESA Online Example

The following example illustrates the job control statements needed to compile and link-edit a DL/I online application written in PL/I for VSE/ESA.

```
// JOB PLISAMPL
// DLBL IJSYSPH,'PL/I TRANSLATION',yyyy/ddd
// EXTENT SYSPCH,extent information
ASSGN SYSPCH,DISK,VOL=volid,SHR
// LIBDEF *,SEARCH=search-libraries
// LIBDEF PHASE,CATALOG=catalog-library
// EXEC DFHEPP1$,SIZE=...
*PROCESS SYSTEM(CICS),XOPTS(CICS,DLI)
        .
        .
   SOURCE DECK
        .
        .
/*
CLOSE SYSPCH,punch
// DLBL IJSYSIN,'PL/I TRANSLATION',yyyy/ddd
// EXTENT SYSIPT
ASSGN SYSIPT,DISK,VOL=volid,SHR
// OPTION NODECK,CATAL
   PHASE PLISAMPL,*
   INCLUDE DFHELII
// EXEC IEL1AA,SIZE=...
// EXEC LNKEDT
/&
// JOB RESET
CLOSE SYSIPT,reader
/&
```

Figure 18. PL/I Online with HLPI or CALL Interface

Note: When the DL/I CALL interface is used the 'DLI' parameter in the *PROCESS XOPTS statement should be omitted.

High Level Assembler Online Example

The following example shows the job control statements needed to assemble and link-edit a DL/I online application written in High Level Assembler for VSE.

```
   // JOB HLASAMPL
   // DLBL   IJSYSPH,'ASSEMBLER TRANSLATION',yyyy/ddd
   // EXTENT SYSPCH,extent-information
   ASSGN SYSPCH,DISK,VOL=volid,SHR
   // LIBDEF *,SEARCH=search-library
   // LIBDEF PHASE,CATALOG=catalog-library
   // EXEC DFHEAP1$,SIZE=...
   *ASM XOPTS(CICS)
   HLASAMPL CSECT
   HLASAMPL AMODE 31
   HLASAMPL RMODE ANY
           .
           .
      SOURCE DECK
           .
           .
   /*
   CLOSE SYSPCH,punch
   // DLBL   IJSYSIN,'ASSEMBLER TRANSLATION',yy/ddd
   // EXTENT SYSIPT
   ASSGN SYSIPT,DISK,VOL=volid,SHR
   // OPTION  SYM,ERRS,NODECK,CATAL
      PHASE   HLASAMPL,*
      INCLUDE DFHEAI
   // EXEC ASMA90,SIZE=...
   // EXEC LNKEDT
   /&
   // JOB RESET
   CLOSE SYSIPT,reader
   /&
```

Figure 19. Example: High Level Assembler - Online

Note: DL/I online applications written in High Level Assembler for VSE must be implemented using the DL/I CALL interface.

DL/I - Batch Applications

For DL/I batch and batch/MPS applications written either in COBOL for VSE/ESA or COBOL II the inclusion of object module DLZBPJRA must be dropped. Therefore the JCL definition for the former entry point CBLCALLA must be replaced by the new entry point DLITCBL.

For DL/I batch and batch/MPS applications written in PL/I for VSE/ESA, the inclusion of IBMBPJRA must be replaced through the inclusion of IBMRPJRA. Also all programs should be compiled with option SYSTEM(DLI).

DL/I batch applications written in High Level Assembler for VSE can be assembled and link-edited in the same way as before.

COBOL Batch Examples

The following examples illustrate the job control statements needed to compile and link-edit a batch or batch/MPS application written either in COBOL for VSE/ESA or COBOL II language. IGYCRCTL is the name for both compilers.

```
// JOB COBSAMPL
// DLBL IJSYSPH,'COBOL TRANSLATION',yyyy/ddd
// EXTENT SYSPCH,extent information
ASSGN SYSPCH,DISK,VOL=volid,SHR
// LIBDEF *,SEARCH=search-libraries
// LIBDEF PHASE,CATALOG=catalog-library
// EXEC DFHECP1$,SIZE=...
 CBL XOPTS(DLI,COBOL2),RES,RENT,DATA(31)
        .
        .
   SOURCE DECK
        .
        .
/*
CLOSE SYSPCH,punch
// DLBL IJSYSIN,'COBOL TRANSLATION',yyyy/ddd
// EXTENT SYSIPT
ASSGN SYSIPT,DISK,VOL=volid,SHR
// OPTION NODECK,CATAL
   PHASE COBSAMPL,*
   INCLUDE DLZLICBL
// EXEC IGYCRCTL,SIZE=...
   ENTRY DLITCBL
// EXEC LNKEDT
/&
// JOB RESET
CLOSE SYSIPT,reader
/&
```

Figure 20. COBOL Batch and Batch/MPS with HLPI Interface

```
// JOB COBSAMPL
// LIBDEF *,SEARCH=search-libraries
// LIBDEF PHASE,CATALOG=catalog-library
// OPTION NODECK,CATAL
   PHASE COBSAMPL,*
// EXEC IGYCRCTL,SIZE=...
 CBL LIB,RES,RENT,DATA(31)
        .
        .
   SOURCE DECK
        .
        .
/*
   ENTRY DLITCBL
// EXEC LNKEDT
/&
```

Figure 21. COBOL Batch and Batch/MPS with CALL Interface

PL/I for VSE/ESA Batch Examples

The following examples illustrate the job control statements needed to compile and link-edit a batch or batch/MPS application written in PL/I for VSE/ESA.

```
// JOB PLISAMPL
// DLBL IJSYSPH,'PL/I TRANSLATION',yyyy/ddd
// EXTENT SYSPCH,extent information
ASSGN SYSPCH,DISK,VOL=volid,SHR
// LIBDEF *,SEARCH=search-libraries
// LIBDEF PHASE,CATALOG=catalog-library
// EXEC DFHEPP1$,SIZE=...
*PROCESS SYSTEM(DLI),XOPTS(DLI)
         .
         .
   SOURCE DECK
         .
         .
/*
CLOSE SYSPCH,punch
// DLBL IJSYSIN,'PL/I TRANSLATION',yyyy/ddd
// EXTENT SYSIPT
ASSGN SYSIPT,DISK,VOL=volid,SHR
// OPTION NODECK,CATAL
   PHASE PLISAMPL,*
// EXEC IEL1AA,SIZE=...
   INCLUDE DLZLIPLI
   INCLUDE IBMRPJRA
   ENTRY DLZLIPLI
// EXEC LNKEDT
/&
// JOB RESET
CLOSE SYSIPT,reader
/&
```

Figure 22. PL/I for VSE/ESA Batch and Batch/MPS with HLPI Interface

```
// JOB PLISAMPL
// LIBDEF *,SEARCH=search-libraries
// LIBDEF PHASE,CATALOG=catalog-library
// OPTION NODECK,CATAL
   PHASE PLISAMPL,*
// EXEC IEL1AA,SIZE=...
*PROCESS SYSTEM(DLI)
         .
         .
   SOURCE DECK
         .
         .
/*
   INCLUDE IBMRPJRA
   ENTRY PLICALLB
// EXEC LNKEDT
/&
```

Figure 23. PL/I for VSE/ESA Batch and Batch/MPS with CALL Interface

Note: The former PL/I ISA storage is now controlled by LE/VSE, which uses GETVIS for storage allocation. This can lead to an increased demand for GETVIS storage of your applications.

High Level Assembler for VSE Batch Example

The following example shows the job control statements needed to assemble and link-edit a DL/I batch application written in High Level Assembler for VSE.

```
// JOB HLASAMPL
// LIBDEF *,SEARCH=search-library
// LIBDEF PHASE,CATALOG=catalog-library
// OPTION CATAL,NODECK
   PHASE HLASAMPL,*
// EXEC ASMA90,SIZE=...
HLASAMPL CSECT
HLASAMPL AMODE 31
HLASAMPL RMODE ANY
         .
         .
   SOURCE DECK
         .
         .
/*
// EXEC LNKEDT
/&
```

Figure 24. Example: High Level Assembler - Batch and MPS Batch

Appendix B. Summary of Customer Interfaces

This appendix contains General-use Programming Interface and Associated Guidance Information, and Product-sensitive Programming Interface and Associated Guidance Information.

DL/I Macros Intended for Customer Use

Table 6 on page 113 identifies the DL/I macros that are provided to allow a customer installation to write programs that use the services of DL/I. Only the macros identified in the figure should be used to request or receive the services of DL/I.

Table 6. Summary of Customer Interfaces

Macro Name	General Use	Product Sensitive	Described in (see list below)
ACCESS	x		RDU, IRDU, GNU
CALLDLI	x		DIAG, CALL
DATASET	x		RDU, IRDU, GNU
DBD	x		RDU, IRDU, GNU, DIAG
DBDGEN	x		RDU, IRDU, GNU, DIAG
DLZACT	x		RDU, IRDU, GNU, DIAG, DBA
DLZBFFR		x	DIAG
DLZBFPL		x	DIAG, DBA
DLZCTRL		x	RDU, RELG
DLZDIB	x		CALL
DLZHDC10-40		x	DBA
DLZMDLI		x	DBA
DLZNN	x		LLC
DLZNNICT	x		LLC
DLZSTBF		x	RELG
DLZTRACE	x		DIAG
DLZTXIT0		x	DIAG
DLZUIB	x		CALL
FIELD	x		RDU, IRDU, GNU
FINISH	x		RDU, IRDU, GNU
LCHILD	x		RDU, IRDU, GNU
PCB	x		RDU, IRDU, GNU
PSBGEN	x		RDU, IRDU, GNU
SEGM	x		RDU, IRDU, GNU
SENFLD	x		RDU, IRDU, GNU
SENSEG	x		RDU, IRDU, GNU

Table 6. Summary of Customer Interfaces (continued)			
Macro Name	**General Use**	**Product Sensitive**	**Described in (see list below)**
VIRFLD	x		RDU, IRDU, GNU
XDFLD	x		RDU, IRDU, GNU

Abbreviation
Publication

CALL
DL/I Application Programming: CALL and RQDLI Interfaces

DBA
DL/I Data Base Administration

DIAG
DL/I Diagnostic Guide

GNU
DL/I DOS/VS User's Guide

IRDU
DL/I Interactive Resource Definition and Utilities

RELG
DL/I Release Guide

RDU
DL/I Resource Definition and Utilities

Related IBM Publications

DL/I VSE 1.12

The latest information on DL/I, is described in the DL/I VSE 1.12.1 Release Guide (this publication).

The following lists all the publications of the DL/I library. In the list below, "(1)" means that the publication is referenced in this publication.

DL/I General Information, GH20-1246
DL/ Library Guide and Master Index, GH24-5008
DL/ Licensed Program Specifications, GH24-5031
DL/I Release Guide, SC33-6211
DL/I DOS/VS User's Guide, SH24-5001 (1)
DL/I Application and Data Base Design, SH24-5022
DL/I Data Base Administration, SH24-5011 (1)
DL/I Resource Definition and Utilities, SH24-5021 (1)
DL/I Interactive Resource Definition and Utilities, SH24-5029 (1)
DL/I Application Programming: HLPI, SH24-5009 (1)
DL/I Application Programming: CALL and RQDLI Interfaces, SH12-5411 (1)
DL/I Messages and Codes, SC34-2641 (1)
DL/I Diagnostic Guide, SH24-5002 (1)
DL/I Recovery/Restart Guide, SH24-5030
IBM System/370 LLC/CC in DL/I DOS/VS Program Reference and Operations Manual, SH20-9046
DL/I Reference Summary: Application Programming, SX24-5103
DL/I Reference Summary: System Programming, SX24-5104
DL/I Reference Summary: High Level Programming Interface, SX24-5120
DL/I Logic, Volume 1, LY12-5016
DL/I Logic, Volume 2, LY12-5015
DL/I Logic Extensions, LY33-9123 (1)

Selected VSE Publications

z/VSE Planning
z/VSE Installation
z/VSE Operation
z/VSE System Control Statements
z/VSE Extended Assressability

Notices

This information was developed for products and services offered in the US. This material might be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

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 grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

For license inquiries regarding double-byte character set (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.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

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 jurisdictions 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.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you provide 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 which has been exchanged, should contact:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

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 document 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.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

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 actual people or business enterprises 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 the 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. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Programming Interface Information

This publication documents intended Programming Interfaces that allow the customer to write programs to obtain services of z/VSE.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml.

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

IPv6/VSE is a registered trademark of Barnard Software, Inc.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Linux is a registered 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.

Terms and Conditions for Product Documentation

Permissions for the use of these publications are granted subject to the following terms and conditions.

Applicability

These terms and conditions are in addition to any terms of use for the IBM website.

Personal use

You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of IBM.

Commercial use

You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM.

Rights

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein. IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed. You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

Accessibility

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use software products successfully. The major accessibility features in z/VSE enable users to:

- Use assistive technologies such as screen readers and screen magnifier software
- Operate specific or equivalent features using only the keyboard
- Customize display attributes such as color, contrast, and font size

Using Assistive Technologies

Assistive technology products, such as screen readers, function with the user interfaces found in z/VSE. Consult the assistive technology documentation for specific information when using such products to access z/VSE interfaces.

Documentation Format

The publications for this product are in Adobe Portable Document Format (PDF) and should be compliant with accessibility standards. If you experience difficulties when you use the PDF files and want to request a web-based format for a publication, you can either write an email to s390id@de.ibm.com, or use the Reader Comment Form in the back of this publication or direct your mail to the following address:

```
IBM Deutschland Research & Development GmbH
Department 3282
Schoenaicher Strasse 220
D-71032 Boeblingen
Federal Republic of Germany
```

In the request, be sure to include the publication number and title.

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Glossary

Listed below are the common terms, abbreviations, and acronyms used throughout this document with a brief definition and/or explanation.

Numerals

31-bit addressing
Provides addressability for address spaces of up to 2 gigabytes.

A

ACB
DL/I application control block, created by the output of DBDGEN and PSBGEN.

ACT
DL/I application control table.

addressing mode (AMODE)
A program attribute that refers to the address length that a program is prepared to handle on entry. Addresses may be either 24 bits or 31 bits in length. In 24-bit addressing mode, the processor treats all virtual addresses as 24-bit values; in 31-bit addressing mode, the processor treats all virtual addresses as 31-bit values.

AMODE
See *addressing mode*.

APAR
Authorized Programming Analysis Report.

C

CICS
Customer Information Control System, a licensed IBM program for online environments.

CICS TS (Customer Information Control System Transaction Server)
An IBM program that controls online communication between terminal users and a database. Transactions entered at remote terminals are processed concurrently by user-written application programs. The program includes facilities for building, using, and servicing databases.

CLC
Component level code.

CMS
Conversational Monitoring System.

Configuration
The combined hardware and software products that make up a data processing system.

CPU
Central processing unit of the computer hardware system.

D

DA
DL/I Documentation Aid.

DATABASE 2
An IBM relational database management system.

DB2
See DATABASE 2.

DBD
Database description.

DB/DC
Database/data communication.

DL/I
Data Language/One, a licensed IBM program.

F

FBA disk device
A fixed block architecture storage device for data in blocks of fixed size; these blocks are addressed by block number relative to the beginning of the file.

FCT
CICS file control table.

G

GSCD
Get system contents directory.

CSD (CICS system definition) file
A component of CICS resource definition online (RDO). It keeps a permanent record of resource information, independently of the active CICS system. The information held in the CSD is used for installing new resources and at a CICS restart.

GSTA
Get statistics.

H

HD
Hierarchical direct, a DL/I access method.

HDAM
Hierarchical Direct Access Method.

HIDAM
Hierarchical Indexed Direct Access Method.

HISAM
Hierarchical Indirect Sequential Access Method.

HLPI
High Level Programming Interface, a DL/I function that allows the COBOL and PL/I Optimizer application programmer to process DL/I databases in batch, MPS batch, and CICS online environments.

HSAM
Hierarchical sequential access method, a DL/I access method.

I

IMF
Interactive Macro Facility, a DL/I function that allows the user to generate DBDs, PSBs, etc. from menu-driven display panels.

IMS/VS
Information Management System/Virtual Storage.

IPF
Interactive Productivity Facility.

IPL
Initial program load.

ISPF
Interactive System Productivity Function, a licensed IBM program required for the use of DL/I IMF and IUG functions. It is the dialog manager for interactive applications.

ISQL
Interactive Structured Query Language.

IUG
Interactive Utilities Generation, a DL/I function that allows the user to generate utility job streams from menu driven display panels.

J

JCL
job control language

M

MPS
Multiple Partition Support.

MSHP
Maintain System History Program.

P

PCB
Program communication block.

PCT
CICS program control table.

PPT
CICS processing program table.

PSB
Program specification block.

PST
Partition specification table.

PTF
Program temporary fix.

R

residency mode (RMODE)
A program attribute that refers to the location where a program is expected to reside in virtual storage.

RMODE
See *residency mode*.

S

SDL
System directory list.

SHSAM
Simple Hierarchical Sequential Access Method.

SLC
Storage Layout Control table; for use in an online environment to specify in which sequence DL/I phases are to be loaded from the library during DL/I initialization.

SPE PTF
Small program enhancement PTF.

SQL/DS
Structured Query Language/Data System.

SVA
Shared virtual area, located in the highest address range of virtual storage. It can contain a system directory list (SDL) of often used phases, resident programs that can be shared between partitions, and an area for dynamic allocation to components of VSE.

System History File
Part of the IBM-distributed VSE system and maintained under the file name IJSYSHF on the (preferred) logical unit SYSREC. This file normally contains all system history status information (product numbers, component IDs, CLC numbers, PTF and APAR numbers, and so on) and is updated by MSHP.

U

UPSI
User program switch indicator.

V

virtual disk
A range of up to two gigabytes of contiguous virtual storage addresses that a program can use as workspace. Although the virtual disk exists in storage, it appears as a real FBA disk device to the user program. All I/O operations directed to a virtual disk are intercepted and the data to be written to, or read from, the disk is moved to or from a data space.

Like a data space, a virtual disk can hold only user data; it does not contain shared areas, system data or programs. Unlike an address space or a data space, data is not directly addressable on a virtual disk. To manipulate data on a virtual disk, the program has to perform I/O operations.

VM/SP
Virtual Machine/System Product.

VS
Virtual Storage.

VSE
Virtual storage extended: A system that consists of a basic operating system (VSE/Advanced Functions) and any IBM supplied and user-written programs required to meet the data processing needs of a user. Its current version is called z/VSE.

VSE/ESA
VSE/Enterprise Systems Architecture.

VSE/VSAM
VSE/Virtual Storage Access Method: the main access method, for direct or sequential processing of fixed and variable length records on disks.

X

XPCC
Cross-partition communication control.

Z

z/VSE
The most advanced VSE system currently available.

Index

A

accessibility 121
allocation of DL/I resources above 16 MB 58
APAR enhancements 7, 61

B

Backout Inflight Changes – CICS Emergency Restart 71
Backout per Database in Batch (PMxxxxx) 6
batch applications 109

C

changed messages 93
CICS Recovery 71
CICS/XRF environment 40
COBOL for VSE/ESA 52
COBOL II 52
Coexistence Environment 58
conditional SVA loading disabled 61
customizing DL/I 19

D

DBD definition, example 48
DBD810 94
DBD811 94
DBD812 94
DFHECI 107
DFHELII 107
DGEN205 95
DGEN258 95
disability 121
DL/I Batch Program Continued Execution 73
DL/I Batch Program Terminated Abnormally 73
DL/I Batch Recovery Process I 74
DL/I Batch Recovery Process II 76
DL/I Batch Recovery Process III 79
DL/I Batch Recovery Process IV 82
DL/I Batch Recovery Process V 83
DL/I Batch Recovery Process VI 86
DL/I Batch Recovery Process VII 88
DL/I Batch Recovery Processes 74
DL/I DOS/VS 1.10 enhancements 52
DL/I Forward Recovery 70
DL/I GETVIS storage in separate subpools 60
DL/I in a VM environment 9
DL/I Recovery 72
DL/I Utility Improvements for Database Recovery (PH29950) 5
DL/I VSE 1.11 enhancements 61
DL/I VSE 1.12.0 enhancements 7
DL/I VSE 1.12.0 features 58
DL/I VSE 1.12.1 features 1, 3–5
DLIOER=ABEND 70
DLIOER=CONTINUE 71
DLZ002I 96
DLZ004I 97
DLZ004I / DLZ005I in a DLI Batch Environment 73
DLZ005I 97
DLZ015I 97
DLZ017I 98
DLZ019I 98
DLZ029I 98
DLZ035I 98
DLZ037I 98
DLZ050I 99
DLZ051I 99
DLZ053I 99
DLZ054I 99
DLZ058I 99
DLZ093I 100
DLZ094I 100
DLZ096I 100
DLZ1028 95
DLZ1055 95
DLZ1059 95
DLZ1060 95
DLZ1061 95
DLZ1062 95
DLZ1064 95
DLZ1065 96
DLZ1066 96
DLZ107I 101
DLZ140I 101
DLZ141I 101
DLZ142A 101
DLZ143I 101
DLZ144I 101
DLZ145I 102
DLZ146I 102
DLZ147A 102
DLZ150I 102
DLZ151I 102
DLZ152I 103
DLZ153I 103
DLZ154I 103
DLZ155I 103
DLZ156I 103
DLZ250I 103
DLZ251I 103
DLZ252 104
DLZ379I 104
DLZ425I 104
DLZ428I 104
DLZ429I 105
DLZ431I 105
DLZ460I 105
DLZ461I 105
DLZ462I 105
DLZ464I 105
DLZ465I 105

DLZ833I 106
DLZ885I 106
DLZ886I 106
DLZ887I 106
DLZMPI00 47
DLZRRC00 47
DLZUACB0 block builder utility 48
DLZURGL0 reload utility 48
DLZURGU0 unload utility 48
dynamic partition support 39

E

enhancements introduced via APARs 7, 61
Enhancements Introduced via APARs 5
enhancements, DL/I DOS/VS 1.10 52

F

features introduced with
DL/I DOS/VS 1.10 42
DL/I DOS/VS 1.7 35
DL/I DOS/VS 1.8 37
DL/I DOS/VS 1.9 39
DL/I VSE 1.11 46
DL/I VSE 1.12.0 58
DL/I VSE 1.12.1 1

G

GETVIS, for program isolation storage 52

H

HD buffers, more than 32 52
HD databases, greater 4GB 48
High Level Assembler 52
HS buffers, VSE/VSAM 42

I

index database, increased VSAM blocksize 52
installation dialogs 11
installation tapes 11
installing DL/I 11

M

macros for customer use 113
messages 93
messages, new and changed 7
migrating 7
Migrating to DL/I VSE 1.12.1 7
migration items
CICS Transaction Server for VSE/ESA 57
DL/I VSE 1.11 migration items 56
migrating to DL/I VSE 1.11 and the CICS Transaction Server for VSE/ESA 1.1 56
migration and compatibility considerations 15
migration from DL/I 1.7.0 to 1.7.1 15
migration from DL/I 1.7.1 to 1.8.0 15
migration from DL/I 1.8.0 to 1.9.0 16
migration items *(continued)*
migration from DL/I 1.9.0 to 1.10.0 16
MPS batch job 47
Multiple MPS Systems 47

N

new and changed messages 93
new messages 93

O

online applications 107
Operating in a CICS XRF Environment 72

P

partitioning
data sets 1
loading database 4
migration 4
operation 5
partition definition 4
partition selection 3
segment lengths 3
utilities 5
PL/I for VSE/ESA 52
post installation tasks 27
Prepare the CICS and DL/I Abnormal Termination Log File 70
PSB, above 16 MB 52

R

Recovering from missing or inappropriate DL/I Recovery 72, 73
Recovery after a DLZ004I or DLZ005I Error 69
Responses to Message DLZ142A 92
RestartCICS and DL/I after DL/I Batch Recovery 91

S

shipment of DL/I 11
shipment tapes 11
Start a new Log Cycle and Take new Backups 91
Stop Log Print on Log Sequence Number (PMxxxxx) 5
storage protection for CICS Transaction Server 51
SVA loading, conditional 52
SVA, loading phases into 19

T

tape as logging device 9

U

using a tape as logging device 9

V

virtual disk 44

Y

year 2000 support 52

Z

z/VSE home page xi

Product Number: 5746-XX1

SC33-6211-09