Version 1 Release 1

IBM Z Table Accelerator Implementation Guide

SC28-3232-00

Contents

Figures

Tables

Preface

This guide provides information on how to implement IBM Z IBM Z® Table Accelerator.

Chapter 1, "Implementation Overview," on page 1 will provide you with more details and a road map on the procedures to be followed.

Audience for this Guide

This guide is intended for use by system installers and/or administrators responsible for implementing IBM Z Table Accelerator. The person implementing IBM Z Table Accelerator should be familiar with a programming language like COBOl, PL/I, C or Assembler, compile procedures, the linkage editor and MVS JCL.

What is Covered in this Guide

This guide contains the implementation procedure for the IBM Z Table Accelerator base product (batch interface) as well as the procedures for the implementation of the CICS/TS, IMS/TM, and the VTS (Virtual Table Share) interfaces. Also covered in this guide are parameter settings, delivered parameter defaults, and overrides.

Naming protocol

All executables begin with DKJ for easy identification, a prefix that has been reserved exclusively by IBM for IBM Z Table Accelerator.

Glossary

The following terms are used throughout this guide:

Alternate Index	An Alternate Index is an Index that may be defined for a Data Table. The Alternate Index has an Alternate Index definition (ALT-DEFINITION) that defines the key, organization, and search order. Alternate Indexes are optional, and there is no limit to the number of Alternate Indexes a Data Table may have.
Data Table	A Data Table is the actual raw data. Each Data Table has a table definition (DT-BLOCK) that is used to generate the Index for the Data Table.
Delivered defaults	The defaults that are delivered with the product. Also known as *factory defaults*.
Index	An Index is defined for each Data Table. A Data Table Index is generated dynamically when a table is opened or defined based on the information in the table definition (DT-BLOCK).
Installation defaults	The defaults set at implementation time by an administrator, which may or may not be the same as the delivered defaults. Defined using the DKJOPTGN file. (These defaults may be overridden by an individual application using the TAOPT file.)

Multitasking Batch	An MVS region that implements multitasking by attaching multiple Task Control Blocks (TCBs). This can include a batch job that attaches several subtasks or a transaction processing region that implements multitasking through multiple TCBs.
Table Expansion	Dynamic allocation of space for tables in the TSR when the current space allocated becomes insufficient.
Temporary Table	A temporary table exists only within a TSR, and is created by the DT command (or IA). It is never stored in a library. A temporary table can be distinguished from a library table using the GD command output—if found, a temporary table will show no dataset name.
TSR, Local TSR	Table Space Region. A data space of up to 2G is used by IBM Z Table Accelerator to house tables. The data space is owned by an application in the associated address space. The application uses IBM Z Table Accelerator to access data within the tables.
TSR of record	If there is no VTS name in the TA-SUBSYSTEM field of the TA-PARM when a command is being executed, the TSR of record is the local TSR, otherwise it is the VTS-TSR that has its name in the TA-PARM.
VTS	Virtual Table Share. The term "VTS" refers to the VTS product, which permits TSRs to be shared among applications. These shared TSRs are called VTS-TSRs.
VTS-TSR, Shared TSR	A Virtual Table Share (VTS) Table Space Region (TSR) is a shared TSR, and resides in a shared data space. Applications can access tables within a VTS-TSR, and use the information as if it were within their local TSRs.
VTS Agent	The DKJVAGNT program, which initializes VTS-TSRs in IBM Z Table Accelerator, and then sits idle until the VTS-TSR is to be terminated.

Conventions used in this guide

This guide uses certain conventions to differentiate code and typed commands, to display the names of parameters, and to describe specific version and release levels of the IBM Z Table Accelerator product.

Convention	**Description**
`code examples and commands`	Code examples and commands appear in this type of font: `this is an example of the font.`
MAXNMTAB	Names of parameters appear in upper case simply for ease of reading; actual case used is upper or lower or a mixture.
Version	Following IBM standards, the term *version* refers to a generation of a software product that has significant new code or new functionality. *Version* is a more general term than *release*. For example, *Version 1* includes *Release 1.1* and *Release 1.2,* and is equivalent to *Release 1.x*.

Convention	Description
Release	Following IBM standards, the term *release* refers to a program or set of programs which represent a specific revision to the base version of a software product. For example, *Release 1.1* is a term that is used to identify the first release of *Version 1*. Subsequent releases made available under the Version 1 umbrella, such as *Release 1.2*, will provide additional revisions to the base product.
Modification Level	Following IBM standards, the term *modification level* refers to the application of specific program enhancements and error corrections to the release of a software product. For example, *Release 1.1.3* is at *modification level* 3, and *Release 1.1.0* is at *modification level* 0.
MVS	MVS is a generic term which is used when referring to z/OS and other related IBM operating systems.

Additional IBM Z Table Accelerator references

This guide is one of a series that describe IBM Z Table Accelerator:

- *IBM Z Table Accelerator Implementation Guide*
- *IBM Z Table Accelerator Concepts and Facilities Guide*
- *IBM Z Table Accelerator Batch Utilities Guide*
- *IBM Z Table Accelerator Administration Guide*
- *IBM Z Table Accelerator Programming Guide*
- *IBM Z Table Accelerator Quick Reference Guide*

Chapter 1. Implementation Overview

The Version 1 IBM Z Table Accelerator software is available for z/OS environments running on 64-bit hardware which supports instructions that are provided with the z10-EC processor or higher. The product consists of the IBM ZTable Accelerator nucleus and the batch, CICS, IMS/TM and VTS components. The VTS software provides an environment to share tables across regions, significantly increasing performance. This chapter covers the information needed to install the IBM Z Table Accelerator software and its components.

Implementation Procedures

The procedures that follow will provide you with a road map of the implementation process. This procedure is for implementing IBM Z Table Accelerator after the SMP downloads. The SMP/E installation process for IBM Z Table Accelerator provides all components needed for implementation in the target libraries.

Step 1—Prepare for implementation

Before you begin the implementation process, you should understand the required tasks, and plan your dataset naming conventions. For more information, see "Step 1—Prepare for IBM Z Table Accelerator implementation" on page 3.

Step 2—Implement the IBM Z Table Accelerator PC Server (required)

Go to the IBM Z Table Accelerator PC Server implementation procedure: "Steps to implement the IBM Z Table Accelerator PC Server" on page 5.

Step 3—Implement the IBM Z Table Accelerator Batch Interface (required)

Go to the MVS Batch Interface implementation procedure: "Steps to implement IBM Z Table Accelerator Batch" on page 9.

Step 4—Implement the VTS Interface (optional)

Go to the optional implementation procedure: "Steps to implement the VTS Interface" on page 13.

Step 5—Implement the IBM Z Table Accelerator CICS Interface (optional)

Go to the optional implementation procedure: "Steps to implement the CICS Interface" on page 15.

Step 6—Implement the IBM Z Table Accelerator IMS TM interface (optional)

Go to the optional implementation procedure: "Steps to implement the IMS/TM Interface" on page 21.

Step 7—Post-implementation

At this point, all implementation steps have been completed.

Your next step is to perform configuration, initialization and administration tasks. See the IBM Z Table Accelerator Administration Guide for further information.

Chapter 2. Preparing for Implementation

To prepare for implementing IBM Z Table Accelerator, perform the steps below.

Step 1—Prepare for IBM Z Table Accelerator implementation

Before starting the implementation procedure, you should be aware of the essential and recommended planning steps — see the overview below, and the recommended preparations.

Implementation overview

Here is an overview of the IBM Z Table Accelerator implementation process:

1. For each IBM Z Table Accelerator interface you implement, you will:
   a. set site-specific system naming-convention defaults for:
      1) load library
      2) IBM Z Table Accelerator libraries
      3) sample libraries
   b. tailor the interface by performing assemblies, link-edits, and moving datasets
   c. run a test procedure to verify successful implementation
   d. update your user documentation to reflect local naming-convention defaults

Recommended preparations

Before starting the implementation process, consider the following:

- **Dataset usage** — Most of the datasets supplied with the product are for access by all of your users (load modules would fall into this category). However, some datasets are for use only by the administrator or implementer for the product. The following sections outline the various uses for the datasets that have been installed. If you want to prevent or limit access to any of these procedures, you may choose to place certain members on different libraries, perhaps with different levels of access.
- **Dataset naming conventions** — You will have to plan your site's actual naming conventions, taking into account such things as: site standards, RACF (or other) access rules, and required DASD and VSAM volume locations. However, it may still be valuable to consider using a unique naming convention for IBM Z Table Accelerator.
- **DASD volumes** — you may have to decide on specific DASD volumes to suit your site standards for installed software.
- **Authority considerations** (for example, RACF) — you may also need to control user access to the various IBM Z Table Accelerator datasets. Be aware that the RACF restrictions on IBM Z Table Accelerator libraries do not apply to tables once they are opened in a TSR. IBM Z Table Accelerator provides other mechanisms to control access to tables in a TSR. For more information, see the IBM Z Table Accelerator Administration Guide.
- **Applicability of special notes and restrictions to your site** — what are the special features of your site that need to be considered during the implementation?

IBM Z Table Accelerator Parameters

Tailor the IBM Z Table Accelerator software for optimal performance in both batch and online environments by setting execution-time parameters and switches. The parameters are listed in Appendix A, “IBM Z Table Accelerator run-time options,” on page 23. Chapters 3 through 7 provide details of their application to a particular IBM Z Table Accelerator interface. For usage details, refer to Appendix A, “IBM Z Table Accelerator run-time options,” on page 23.

Default execution-time parameters can be modified by changing the default values or by specifying them at execution time.

To modify the default values for a specific interface, change the DKJOPTGN macro operands in the appropriate member of the SRC dataset, reassemble and relink it. Models for reassembling and relinking are supplied in the CNTL partitioned data set. These procedures are described in the chapters that follow.

At execution time, the default values can be overridden by specifying them in a dataset defined by DDNAME TAOPT. For details on coding parameters in the TAOPT dataset, see the parameters described in Appendix A, "IBM Z Table Accelerator run-time options," on page 23.

Enqueues

Integrity of IBM Z Table Accelerator is achieved by issuing enqueue requests. The major enqueue name is DKJIZTA. The scope of the enqueues is SYSTEMS.

Modifications to Global Resource Serialization (GRS) PARMLIB controls for your installation must not prevent the propagation of the major enqueue name throughout the complex. Failure to adhere to this can result in the corruption of the datasets on DASD shared between several MVS images running IBM Z Table Accelerator.

Since these requirements are beyond the scope of the IBM Z Table Accelerator software, you must ensure that these conditions are met if the same datasets are to be shared among separate copies of the operating system.

Step 2—Verify recommended security considerations

IBM Z Table Accelerator datasets have been installed through SMP/E. The following datasets should be available. Whether the datasets created by the SMP/E process will be directly used or whether copies will be made is up to the installation.

The following datasets should be available for read to everyone, but include members which only an IZTA administrator would execute:

- your.prefix.CNTL
- your.prefix.LOAD
- your.prefix.TASYSLB
- your.prefix.CLIST

If your.prefix.LOAD is not to be APF-authorized, the following dataset must be created and authorized for the PC Server job and the VTS Agent jobs as required:

- your.prefix.AUTHLIB

The following dataset is a sample IBM Z Table Accelerator table library and could be available for write to everyone as it is intended for testing by users of IBM Z Table Accelerator:

- your.prefix.MAINLIB

The following dataset is used only by IBM Z Table Accelerator implementation or administrative personnel:

- your.prefix.SRC

Note: To perform configuration, initialization and administration tasks, see this guide and the IBM Z Table Accelerator Administration Guide.

Chapter 3. Implementing the PC Server

This chapter describes the procedure for implementing the IBM Z Table Accelerator PC Server. The PC Server must be initiated and be running whenever any IBM Z Table Accelerator application is executing on the MVS image.

The PC Server must run from an APF-authorized library and should be running at all times, starting before the first use of IBM Z Table Accelerator in the MVS image. It is recommended that it be set up as a started task and initiated at system initialization.

For more information on the PC Server, see the IBM Z Table Accelerator Administration Guide.

Steps to implement the IBM Z Table Accelerator PC Server

With all members and datasets available from the SMP/E installation, you can now proceed to implement the IBM Z Table Accelerator PC Server on your system. This will consist of several steps:

1. "Step 1—Move load modules to authorized load library" on page 5
2. "Step 2—Modify DKJPCSRV PROC and install in PROCLIB" on page 6
3. "Step 3—Start the PC Server" on page 7

The JCL members for this step can be found in your.prefix.CNTL. To proceed with the implementation, perform the steps below.

Step 1—Move load modules to authorized load library

If your.prefix.LOAD is already authorized, skip this step.

- Modify the CNTL member DKJAUTH to conform to your specific environment (see Figure 1 on page 6). Specifically:
  – dataset names in DD statements IZTA and AUTHLIB
- Execute DKJAUTH and confirm successful return code.

```
//* Insert your job card here
//*
//********************************************************************
//*  Copies base product modules to authlib
//********************************************************************
//*
//IEBCOPY  EXEC PGM=IEBCOPY
//IZTA     DD  DISP=SHR,DSN=your.prefix.IZTA.LOAD           <======
//AUTHLIB  DD  DISP=SHR,DSN=your.prefix.IZTA.AUTHLIB        <======
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
  COPY INDD=((IZTA,R)),OUTDD=AUTHLIB
  S M=(DKJPCSRV,DKJTPCRM)
  S M=(DKJVAGNT,DKJVBASE)
*    COPY PRODUCT API MODULE FOR VTS STARTUP
  S M=(DKJTCALL)
  S M=(DKJTVROT)
*    COPY NUCLEUS AND ROOT MODULES FOR VTS STARTUP
  S M=(DKJTNUCL,DKJTNAME)
  S M=(DKJTROTB,DKJTNTPD)
/*
```

Figure 1. Sample DKJAUTH JCL

Notes:

1. You may require the assistance of a systems programmer for this step.
2. DKJAUTH specifies load member names required by the IBM Z Table Accelerator PC Server (and VTS) that must reside in an authorized load library.

Step 2—Modify DKJPCSRV PROC and install in PROCLIB

The PROC shown in this section is intended to be used as a started task for the PC Server and should be a member in the SYS1.PROCLIB concatenation. It is distributed in your.prefix.CNTL(DKJPCSRV).

A. Modify DKJPCSRV PROC

- Modify the sample PROC DKJPCSRV to conform to your specific environment (see Figure 2 on page 7). Specifically:
  - dataset name in STEPLIB statements

Notes:

1. The STEPLIB DD dataset name originates from those defined earlier. Use the dataset name specified in the AUTHLIB statement of CNTL member DKJAUTH (see "Step 1—Move load modules to authorized load library" on page 5). If Step 1 was skipped because your.prefix.LOAD was an authorized library, use your.prefix.LOAD
2. If you change the member name for DKJPCSRV, ensure that the same name is used to start up the PC Server in the next step.

```
//DKJPCSRV PROC
//*
//*****************************************************
//* Runs the PC (Program Call) Server job
//*****************************************************
//*
//DKJPCSRV EXEC PGM=DKJPCSRV,TIME=1440
//STEPLIB  DD  DISP=SHR,DSN=your.prefix.IZTA.AUTHLIB          <======
//SYSUDUMP DD SYSOUT=*
//*
```

Figure 2. Sample PROC DKJPCSRV JCL

B. Copy the DKJPCSRV PROC member

- Copy the modified DKJPCSRV member to a PROCLIB in the SYS1.PROCLIB concatenation so it can be started with the MVS START command.

Step 3—Start the PC Server

Issue the MVS command

S DKJPCSRV

or, alternatively, create your own job to run PROC DKJPCSRV and submit it.

```
$HASP373 DKJPCSRV STARTED
IEF403I DKJPCSRV - STARTED - TIME=11.11.15
DKJ00800I IZTA PC Server V110 Available
```

Figure 3. Sample DKJPCSRV job output

Chapter 4. Implementing IBM Z Table Accelerator z/OS Batch

This chapter describes basic steps required to implement the IBM Z Table Accelerator batch component.

The IBM Z Table Accelerator batch software is the IBM Z Table Accelerator base product. Implement this component first, then implement any optional components and features.

Steps to implement IBM Z Table Accelerator Batch

To implement the IBM Z Table Accelerator batch interface, perform the following steps:

1. "Step 1—Modify default parameters (optional)" on page 9
2. "Step 2—Copy IBM Z Table Accelerator load modules (if necessary)" on page 10
3. "Step 3—Create IBM Z Table Accelerator table libraries from distributed PDSes (mandatory)" on page 10
4. "Step 4—Update DKJCDRV CLIST to use IBM Z Table Accelerator table libraries" on page 10
5. "Step 5—Update binder procedure to include IBM Z Table Accelerator API stub (if necessary)" on page 10
6. "Step 6—Convert IBM Z Table Accelerator table libraries from BDAM to VSAM (optional)" on page 10
7. "Step 7—Confirm successful batch implementation" on page 11

Step 1—Modify default parameters (optional)

If desired, modify the default parameters (see "IBM Z Table Accelerator Parameters" on page 3) for operating IBM Z Table Accelerator in a batch environment. If the default parameters are acceptable or you plan to use the Override feature provided through the TAOPT dataset, skip this step.

For parameter descriptions, see Appendix A, "IBM Z Table Accelerator run-time options," on page 23. The default values for this component are listed below and are also defined in:

your.prefix.SRC(DKJT1134)

Default batch parameters:

```
DKJOPTGN BASE,
    HASH_HI_DEN_LIM=900,
    HASH_LOW_DEN_LIM=600,
    LISTOPTIONS=N,
    LOCKTIMERC=0,
    LOCKTIMEWTO=30,
    MAXNMTAB=0,
    MTRETAIN=N,
    MULTITASKING=N,
    OVRRIDES=YYNYYYYY,
    RACF_LIBACCESS=N,
    STROBE=0,
    SWITCHES=YNNYNNNN,
    TABLEWAITRC=0,
    TABLEWAITWTO=30,
    TSR_ALGORITHM=P,
    TSR_WARNING_FREQ=1,
    TSR_WARNING_PCT=85,
    TSRSIZE=10M,
    ZEROROWS=Y,
    ML=MAINLIB
```

To make changes to the default batch parameters:

- Modify: your.prefix.SRC(DKJT1134).

- Reassemble and link-edit the module into the IZTA batch interface program DKJBBASE using the following as a model: your.prefix.CNTL(DKJAOPTB)

Step 2—Copy IBM Z Table Accelerator load modules (if necessary)

If it is necessary to copy the IBM Z Table Accelerator load modules to another system library, use the IBM utility IEBCOPY.

Step 3—Create IBM Z Table Accelerator table libraries from distributed PDSes (mandatory)

Tables to be loaded into IBM Z Table Accelerator table libraries are distributed as members of PDSes.

A. IZTA table library contents

The following IBM Z Table Accelerator table libraries and tables will be created:

- your.prefix.TASYSLB (from PDS your.prefix.DKJSLIB)
  - TADRHELP (from member DKJHELP)
  - TADRMSG (from member DKJMSG)
  - TADRTUTR (from member DKJTUTR)
- your.prefix.MAINLIB (from PDS your.prefix.DKJMLIB)
  - EXAMPLE (from member DKJXMPL)
  - EXAMALT (defined by program DKJTEXEC)

B. IZTA table library dataset creation

The following steps require "Step 3—Start the PC Server" on page 7 to have been successful and the IBM Z Table Accelerator PC Server be running on the same MVS image.

- Modify and execute:

  - your.prefix.CNTL(DKJCSLB) to define and populate your.prefix.TASYSLB
  - your.prefix.CNTL(DKJCMLB) to define and populate your.prefix.MAINLIB

Step 4—Update DKJCDRV CLIST to use IBM Z Table Accelerator table libraries

This CLIST allows a TSO user to access IBM Z Table Accelerator interactively.

- Customize the REXX CLIST in your.prefix.CLIST(DKJCDRV). Make changes to the following:

```
loadlib = '**your.prefix**.LOAD'
slib = '**your.prefix**.TASYSLB'
```

Step 5—Update binder procedure to include IBM Z Table Accelerator API stub (if necessary)

If all programs will call DKJTCALL dynamically, this step is not needed. Otherwise, to satisfy static calls to DKJTCALL, your application program binder procedure will need to add your.prefix.LOAD to its SYSLIB concatenation.

Step 6—Convert IBM Z Table Accelerator table libraries from BDAM to VSAM (optional)

All IBM Z Table Accelerator table libraries that have been created in "Step 3—Create IBM Z Table Accelerator table libraries from distributed PDSes (mandatory)" on page 10 have been defined with a file organization of BDAM.

If you wish to convert these libraries to VSAM file organization, customize and run the job in your.prefix.CNTL(DKJB2V) member for each IBM Z Table Accelerator table library to be converted.

Note:

1. Libraries do not have to be converted to VSAM.
2. You may have a mixture of BDAM and VSAM libraries.
3. For VSAM library definition information, see the IBM Z Table Accelerator Administration Guide chapter on “Best Practices”.

Step 7—Confirm successful batch implementation

Ensure that the IBM Z Table Accelerator PC Server has been installed and is running on the same LPAR (MVS image) as IBM Z Table Accelerator. If the PC Server is not running at the time that an application accessing IBM Z Table Accelerator software initializes, the job which attempted to access IBM Z Table Accelerator will abend and you will see the following error message in the JES LOG:

```
DKJ00201S IZTA PC Server unavailable.
```

To verify IBM Z Table Accelerator batch implementation:

- Modify the statements and names in the batch validation job in your.prefix.CNTL(DKJVDTE1).
- Execute this job and confirm successful return code.

Note: This job tests the command executor utility DKJTDRV and the IBM Z Table Accelerator batch library utility DKJTEXEC. As the job leaves no permanent changes, it may be run at any time to confirm that the base product is operational.

Chapter 5. Implementing the VTS (Virtual Table Share) Interface

This chapter describes the Virtual Table Share (VTS) implementation procedures.

Steps to implement the VTS Interface

For details about the operation of VTS, see the IBM Z Table Accelerator Administration Guide and the IBM Z Table Accelerator Programming Guide.

To proceed with the implementation, perform the following steps:

1. "Step 1—Modify default parameters (optional)" on page 13
2. "Step 2—Copy IBM Z Table Accelerator load modules (optional)" on page 13
3. "Step 3—Confirm successful VTS implementation" on page 14

Step 1—Modify default parameters (optional)

Modify the default parameters for operating the VTS Agent, if desired. If the default values are acceptable, or you wish to use the Override feature provided via the TAOPT dataset, no additional work is required in this step.

For parameter descriptions, see Appendix A, "IBM Z Table Accelerator run-time options," on page 23. The default values for this product are listed below and are also defined in:

your.prefix.SRC(DKJV1134).

Default VTS parameters

```
DKJOPTGN VTS,
    LISTOPTIONS=N,
    MAXNMTAB=0,
    MTRETAIN=N,
    STROBE=0,
    STROBEMETHOD=0,
    TSR_ALGORITHM=P,
    TSR_WARNING_FREQ=1,
    TSR_WARNING_PCT=85,
    TSRSIZE=10M,
    ZEROROWS=Y,
    VTSNAME=DKJA
```

To make changes to the default parameters:

- Modify your.prefix.SRC(DKJV1134).
- Reassemble DKJV1134 and link-edit into load module DKJVBASE. Use your.prefix.CNTL(DKJAOPTV) as a model.

Step 2—Copy IBM Z Table Accelerator load modules (optional)

The VTS Agent must be run from an APF-authorized library. In "Step 1—Move load modules to authorized load library" on page 5 the load modules required by the VTS Agent were copied to an authorized library. If your.prefix.LOAD is not APF-authorized, then use the authorized library used by the DKJPCSRV started Procedure. A system programmer should be able to confirm that an appropriate authorized load library is set up.

Step 3—Confirm successful VTS implementation

If the IBM Z Table Accelerator PC Server is not running at the time that IBM Z Table Accelerator VTS software initializes, the VTS job which attempts to access IBM Z Table Accelerator will abend and the following error message will be displayed in the JES LOG:

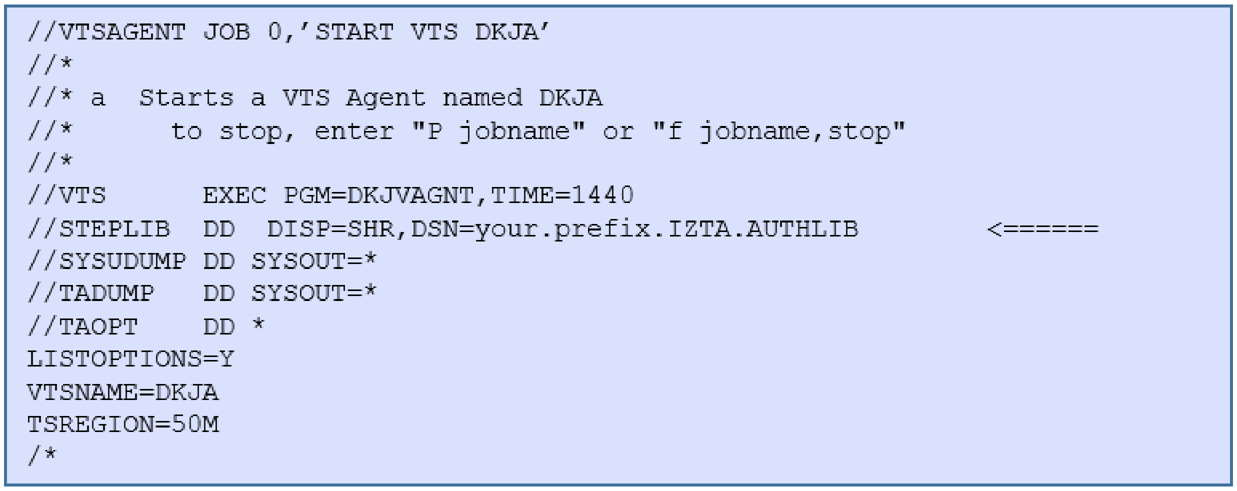
```
DKJ00605E V110 Environment not ready; VTS stopping
```

To verify that VTS is implemented correctly, perform the following steps:

A. Modify the VTS Agent Job

- Modify the job in your.prefix.CNTL(DKJVTSJ) for your environment. Specifically:

    - dataset name in STEPLIB statement



```
//VTSAGENT JOB 0,'START VTS DKJA'
//*
//* a  Starts a VTS Agent named DKJA
//*      to stop, enter "P jobname" or "f jobname,stop"
//*
//VTS      EXEC PGM=DKJVAGNT,TIME=1440
//STEPLIB  DD  DISP=SHR,DSN=your.prefix.IZTA.AUTHLIB        <======
//SYSUDUMP DD SYSOUT=*
//TADUMP   DD SYSOUT=*
//TAOPT    DD *
LISTOPTIONS=Y
VTSNAME=DKJA
TSREGION=50M
/*
```

Figure 4. Modify DKJVTSJ

B. Ensure that the IBM Z Table Accelerator PC Server is up

- Ensure that the PC Server is running on the same LPAR (MVS image).

C. Execute VTS Agent job

- Execute job DKJVTSJ and confirm successful return code.

  A VTS Agent runs until it is explicitly stopped by an MVS P or F command

  Check the JES message log for the following message from the VTS job:

  ```
  DKJ00600I IZTA V110 VTS DKJA initialized.
  ```

D. Modify and execute the job for accessing the VTS

- Modify and execute job in your.prefix.CNTL(DKJVDTE2)

Note: this job does not require an authorized library

E. Stop the VTS Agent

- Stop the VTS-TSR using MVS command:

  ```
  P jobname
  ```

Chapter 6. Implementing the CICS Interface

This chapter describes the implementation procedures for the IBM Z Table Accelerator CICS interface.

Steps to implement the CICS Interface

To implement the IBM Z Table Accelerator CICS Interface, perform the following steps:

1. "Step 1—Modify default parameters (optional)" on page 15
2. "Step 2—Copy IBM Z Table Accelerator load modules (optional)" on page 16
3. "Step 3—Convert from BDAM table libraries to VSAM (optional)" on page 16
4. "Step 4—Create CICS CSD definitions for IBM Z Table Accelerator table libraries" on page 16
5. "Step 5—Create CICS CSD definitions for IBM Z Table Accelerator programs and transactions" on page 16
6. "Step 6—Create CICS definitions for IBM Z Table Accelerator journal file" on page 17
7. "Step 7—Add IBM Z Table Accelerator entries to PLT" on page 17
8. "Step 8—Update binder procedure to include IBM Z Table Accelerator API stub (if necessary)" on page 17
9. "Step 9—Customize CICS startup JCL" on page 17
10. "Step 10—Confirm successful CICS Interface implementation" on page 18
11. "Step 11—Change system abends to task abends" on page 18

Step 1—Modify default parameters (optional)

Modify the default parameters for operating IBM Z Table Accelerator in a CICS environment, if desired. If the default values are acceptable or you wish to use the Override feature provided via the TAOPT dataset, no additional work is necessary in this step.

For parameter descriptions, see Appendix A, "IBM Z Table Accelerator run-time options," on page 23. The default values for this component are listed below and are also defined in:

your.prefix.SRC(DKJT2734)

Default CICS parameters

```
DKJOPTGN CICS,
    CICSJRNL=99,
    HASH_HI_DEN_LIM=900,
    HASH_LOW_DEN_LIM=600,
    LISTOPTIONS=N,
    LOCKTIMERC=0,
    LOCKTIMEWTO=30,
    MAXNMTAB=0,
    MTRETAIN=N,
    OVRRIDES=YNNYYYYY,
    SUPRESS_DUMPS=NONE,
    STROBE=0,
    SWITCHES=YNNYNNNN,
    TABLEWAITRC=0,
    TABLEWAITWTO=30,
    TSR_ALGORITHM=P,
    TSR_WARNING_FREQ=1,
    TSR_WARNING_PCT=85,
    TSRSIZE=10M,
    ZEROROWS=Y,
    ML=MAINLIB
```

To make changes to the default parameters:

- Modify your.prefix.SRC(DKJT2734).

- Reassemble and link-edit the module into the IBM Z Table Accelerator CICS interface program DKJCBASE. Use your.prefix.CNTL(DKJAOPTC) as a model.

Step 2—Copy IBM Z Table Accelerator load modules (optional)

If it is necessary to copy the IBM Z Table Accelerator load modules to another system library, use the IBM utility IEBCOPY to do so.

Step 3—Convert from BDAM table libraries to VSAM (optional)

If you are converting any of the IBM Z Table Accelerator table libraries from the installation or any of your own table libraries from a BDAM to a VSAM file organization, see "Step 6—Convert IBM Z Table Accelerator table libraries from BDAM to VSAM (optional)" on page 10 for information to perform this task.

Step 4—Create CICS CSD definitions for IBM Z Table Accelerator table libraries

Create CICS CSD resource definitions for each IBM Z Table Accelerator table library accessed under CICS:

- member DKJCFCT in your.prefix.SRC contains macro instructions for defining BDAM files
- member DKJ110V in your.prefix.SRC contains CICS System Definition (CSD) resource definitions for defining VSAM files

In addition to MAINLIB, you need to include entries for additional IBM Z Table Accelerator table libraries to be referenced under CICS.

Define all IBM Z Table Accelerator table libraries in CICS as NON RECOVERABLE. This is accomplished with LOG=NO on the FCT for any IBM Z Table Accelerator table library or RECOVERY(NONE) on the CSD.

Note:

1. Ensure that any IBM Z Table Accelerator BDAM table libraries are defined using the DFHFCT macros. BDAM datasets cannot be defined as resource definitions in the CSD dataset.
2. IBM Z Table Accelerator software has been designed to ensure the integrity of its table libraries, so there is no need to use CICS back out and recovery mechanisms. If CICS is permitted to perform back out and recovery on IBM Z Table Accelerator libraries, a transaction interlock occurs. In addition, IBM Z Table Accelerator libraries may be damaged during the CICS recovery process.

Step 5—Create CICS CSD definitions for IBM Z Table Accelerator programs and transactions

Create CICS definitions for each IBM Z Table Accelerator program and transaction accessed under CICS.

If DKJTCALL is called dynamically with a CICS link, and auto-install (SIT:PGAIPGM=ACTIVE) is not used then DKJTCALL must be defined in the CSD. If DKJTCALL is called through MVS, then MVS will resolve the calls and the CICS definition is not required.

The member DKJ110 in your.prefix.SRC contains the CSD instructions for defining IBM Z Table Accelerator programs and transactions to CICS.

Note:

1. It is important that the modules marked resident remain marked resident; this is part of the IBM Z Table Accelerator architecture and any changes will result in product failure. These are DKJTNAME, DKJTNUCL, DKJTROTC and DKJCBASE.
2. Programs DKJTCIN and DKJTSHUT must be defined as quasi-reentrant on the CONCURRENCY attribute of the program resource definition.
3. IBM Z Table Accelerator programs must be defined in the CSD; they cannot be auto-installed. This is because some IBM Z Table Accelerator programs are loaded by the PLTPI program, DKJTCIN, using EXEC CICS SET...PHASEIN and this is not supported by auto-install.

Step 6—Create CICS definitions for IBM Z Table Accelerator journal file

Create the CICS definition for the IBM Z Table Accelerator journal file specified when customizing DKJT2734. The default is 99.

The CICS journal functions are performed in conjunction with the MVS logger. Allocate the necessary log streams on the MVS logger for the journal file specified in DKJT2734. Define the log stream for the journal file in CICS in the CSD.

The member DKJCSD99 in your.prefix.SRC contains sample instructions for defining the MVS log stream for the CICS journal file.

Step 7—Add IBM Z Table Accelerator entries to PLT

If you have a PLTPI member in your CICS startup, edit the member in the Program List by entering DKJTCIN in the line after the entry for PROGRAM=DFHDELIM. If not, use the sample in member DKJPLTI in your.prefix.SRC as a guide.

Similarly, if you have a PLTSD member in your CICS definition, edit the member in the Program List by entering DKJTSHUT in the line after the entry for PROGRAM=DFHDELIM. If not, use the sample in member DKJPLTT in your.prefix.SRC as a guide.

During system startup this entry causes the IBM Z Table Accelerator Resource Manager Interface to initialize. If the initialization is not performed or is not successful (required programs are not defined or not available), then all CICS tasks using DKJTCALL will terminate with abend code AEY9.

Step 8—Update binder procedure to include IBM Z Table Accelerator API stub (if necessary)

If all programs will call DKJTCALL dynamically, this step is not needed. Otherwise, to satisfy static calls to DKJTCALL, your application program binder procedure will need to add your.prefix.LOAD to its SYSLIB concatenation.

Step 9—Customize CICS startup JCL

Customize your CICS start-up JCL (sample in your.prefix.CNTL(DKJCICSJ) to include:

- access to the IBM Z Table Accelerator load library your.prefix.LOAD (or the load library to which the modules have been moved to)
- the IBM Z Table Accelerator table library "your.prefix.TASYSLB"
- any other IBM Z Table Accelerator libraries used by your online applications
- If the TAOPT dataset is to be used, see the CICS chapter in the IBM Z Table Accelerator Administration Guide.

Note: All IBM Z Table Accelerator table libraries may be allocated with DISP=SHR.

To reference an existing IBM Z Table Accelerator table library, use this JCL as a pattern:

```
//DDNAME DD DSN=your.prefix.MAINLIB,DISP=SHR
```

Each IBM Z Table Accelerator table library is identified by a unique DDNAME. You may not associate more than one DDNAME with the same IBM Z Table Accelerator table library. Multiple libraries cannot be concatenated using a single DD statement. The IBM Z Table Accelerator ML command is available for logically concatenating libraries.

Step 10—Confirm successful CICS Interface implementation

Ensure that the IBM Z Table Accelerator PC Server is executing on the same LPAR (MVS image). If it is not, the first access to IBM Z Table Accelerator in the region will abend and you will see the following error message in the JES message log:

```
DKJ00567E The IZTA PC Server must be running for IZTA to initialize.
```

Once the CICS region has been successfully started, verify the implementation of the IBM Z Table Accelerator CICS interface.

- Sign on to CICS.
- On a clear screen enter the transaction TADR.
- Press <ENTER>. The IBM Z Table Accelerator CICS Driver screen is displayed. You are now able to enter IBM Z Table Accelerator and DRIVER commands.
- On the first line of the IBM Z Table Accelerator Driver screen, enter the command PR and the table name EXAMPLE.
- Press <ENTER>. The data is displayed on the screen if the IBM Z Table Accelerator table library your.prefix.MAINLIB has been allocated to DDname MAINLIB in the CICS region startup JCL.
- On the first line of the screen, try a BN command. The product name and release level will be displayed. This should confirm that you have successfully installed the IBM Z Table Accelerator CICS interface.

For more details on the operation of the IBM Z Table Accelerator CICS DRIVER, see the IBM Z Table Accelerator Programming Guide.

If the CICS implementation was successful pressing <PF3> terminates the CICS DRIVER; if the implementation was unsuccessful contact Technical Support for help.

Step 11—Change system abends to task abends

This step is recommended for any CICS regions that access a IBM Z Table Accelerator VTS-TSR. If the VTS-TSR region is cancelled or abends there is a chance that CICS tasks would fail. The failure is displayed (or listed) as an MVS S01D or S067 abend. These two abends bring down the entire CICS region. They can be converted to Task (rather than System) abends by updating the CICS region System Recovery table (SRT).

- Review the SRT information in the CICS documentation section of the IBM manual "Resource Definition Guide" to determine if it is possible to revise the SRT at your site.
- Check if the abend codes are already in the SRT used by your CICS region’s start-up parameters (SIT). If the abend codes are present do not continue.
- If the abend code is not present and you wish to add the codes then:
  - Alter the job DKJCSRT in your.prefix.CNTL to meet your installation's requirements (see Figure 5 on page 19).
  - Submit the job to assemble and link the new SRT containing the entries S01D and S067.
  - Alter the start-up parameters (SIT) for each CICS region to use the revised SRT.

```
//* Insert your job card here
//*
//* Sample job for converting system abends S01D and S067 to task abends
//*    in the SRT
//*
//DFHAUPLE EXEC DFHAUPLE,NAME=SDFHLOAD
//ASSEM.SYSUT1 DD *
TITLE 'IZTA CICS SYSTEM RECOVERY TABLE'
PRINT NOGEN
DFHSRT TYPE=INITIAL,SUFFIX=xx
DFHSRT TYPE=SYSTEM,RECOVER=YES,ABCODE=(01D,067)
ABCODE=(01D,067)
DFHSRT TYPE=FINAL
END
/*
```

Figure 5. Sample SRT job

Chapter 7. Implementing the IMS TM Interface

This chapter describes the implementation procedures for the IBM Z Table Accelerator IMS interface.

Steps to implement the IMS/TM Interface

To implement the IBM Z Table Accelerator IMS/TM Interface, perform the following steps:

1. "Step 1—Modify default parameters (optional)" on page 21
2. "Step 2—Copy IBM Z Table Accelerator load modules (optional)" on page 21
3. "Step 3—Check pre-load list" on page 21
4. "Step 4—Customize IMS startup JCL" on page 22
5. "Step 5—Confirm successful IMS Interface implementation" on page 22

Step 1—Modify default parameters (optional)

If desired, modify the default parameters for operating the IBM Z Table Accelerator software in an IMS Message Processing Region (MPR). If the default values are acceptable or you wish to use the Override feature provided via the TAOPT dataset, no additional work is necessary in this step.

For parameter descriptions, see Appendix A, "IBM Z Table Accelerator run-time options," on page 23. The default values for this component are listed below and are also defined in your.prefix.SRC(DKJT1334).

Default IMS parameters

```
DKJOPTGN IMS,
    HASH_HI_DEN_LIM=900,
    HASH_LOW_DEN_LIM=600,
    LISTOPTIONS=N,
    LOCKTIMERC=0,
    LOCKTIMEWTO=30,
    MAXNMTAB=0,
    MTRETAIN=N,
    OVRRIDES=YYNYYYYY,
    STROBE=0,
    SWITCHES=NNNYNNNN,
    TABLEWAITRC=0,
    TABLEWAITWTO=30,
    TSR_ALGORITHM=P,
    TSR_WARNING_FREQ=1,
    TSR_WARNING_PCT=85,
    TSRSIZE=10M,
    ZEROROWS=Y,
    ML=MAINLIB
```

To make changes to the default parameters:

- Modify your.prefix.SRC(DKJT1334).
- Reassemble and link-edit the module into the IBM Z Table Accelerator IMS interface program DKJIBASE using your.prefix.CNTL(DKJAOPTI) as a model.

Step 2—Copy IBM Z Table Accelerator load modules (optional)

If it is necessary to copy the IBM Z Table Accelerator load modules to another system library, use the IBM utility IEBCOPY to do so. If re-blocking is required use the COPYMOD control statement.

Step 3—Check pre-load list

At this point, you should ensure that your IMS pre-load list is updated. See the IMS Chapter in the IBM Z Table Accelerator Administration Guide.

Step 4—Customize IMS startup JCL

Review the information in the IBM Z Table Accelerator Administration Guide for the JCL for IMS/TM transactions (rules for IBM Z Table Accelerator batch apply to BMPs and IMS Batch), pre-loading of IBM Z Table Accelerator modules, restrictions, etc. Make the changes to your IMS environment that are listed in the IBM Z Table Accelerator Administration Guide.

Step 5—Confirm successful IMS Interface implementation

Ensure that the IBM Z Table Accelerator PC Server is running on the same LPAR (MVS image) as this interface. If the PC Server is not running at the time that the IBM Z Table Accelerator software initializes, the transaction which issues the IBM Z Table Accelerator call will abend and you will see the following error message in the JES message log:

```
DKJ00201S IZTA PC Server unavailable.
```

To verify successful implementation of the IMS TM interface:

- Prepare or modify a simple IMS transaction to include a call to DKJTCALL, for example, use an LS command to return the current status switch settings.
- Prepare a TAOPT file with at least this parameter — LISTOPTIONS=Y.
- Try your transaction.
- Examine the MPR's JES MSG log using, for example, SDSF for various start-up messages from IBM Z Table Accelerator that show the active runtime options. This is an indication that IBM Z Table Accelerator has initialized successfully and is ready for subsequent calls from your transactions.

Appendix A. IBM Z Table Accelerator run-time options

This appendix lists the IBM Z Table Accelerator run-time option parameters. Most occur in all interfaces. If you are running applications across multiple environments any changes to these options in one interface may need to be made in each installed interface. Options can be customized for your site at implementation time, using the DKJOPTGN macro in DKJTxx34 modules. Applications can further override options on an individual basis, using the TAOPT dataset.

CICSJRNL—CICS Journal File ID

This parameter identifies the CICS Journal File onto which the TABLE SPACE Report records are to be written by the system, if strobe records are to be written. The delivered default value for the CICSJRNL parameter is 99.

This option can only be specified for the IBM Z Table Accelerator CICS interface.

This option applies to this region's users, regardless of which TSR they access.

HASH_HI_DEN_LIM—High Density Limit for Hash Indexes

HASH_LOW_DEN_LIM and HASH_HI_DEN_LIM limit the density of the index for a hash table. These values are designed to prevent performance problems which can occur when inappropriately high values are used when defining hash tables (can result in a lot of key collisions in the table). Other problems occur if the difference between low and high density values is too small. A ratio of 2/3 is now enforced: Low density may not be greater than 2/3 of high density.

HASH_HI_DEN_LIM=nnn must be between 100 and 900 (10% and 90%); the default is 900. We recommend the value be lowered to 500.

This option applies to this region's users, regardless of which TSR they access.

This option can be specified for all IBM Z Table Accelerator interfaces except the VTS interface.

HASH_LOW_DEN_LIM—Low Density Limit for Hash Indexes

See above ("HASH_HI_DEN_LIM—High Density Limit for Hash Indexes" on page 23).

HASH_LOW_DEN_LIM=nnn must be between 10 and 600; the default is 600. We recommend the value be lowered to between 200 and 300.

This option applies to this region's users, regardless of which TSR they access.

This option can be specified for all IBM Z Table Accelerator interfaces except the VTS interface.

LIBnn, ML—IBM Z Table Accelerator Library List

Specify in sequence, the names of the libraries (DDNAMEs) in the IBM Z Table Accelerator Library List (LIB-LIST). In DKJOPTGN, this is specified as a comma delimited list under ML. In the TAOPT file, the library names are specified as LIBnn=lib_name. Up to 10 LIBnn may be specified, as long as the numbers are contiguous. For example, LIB01=MAINLIB, LIB02=TESTLIB.

LIBnn numbers must be contiguous AND they must start with LIB01 or LIB02, otherwise IBM Z Table Accelerator initialization will fail.

The delivered default is ML=MAINLIB which is equivalent to supplying a single parameter LIB01=MAINLIB. If no ML is set, IBM Z Table Accelerator uses MAINLIB.

This option applies to this region's users, regardless of which TSR they access.

This option can be specified for all IBM Z Table Accelerator interfaces.

LISTOPTIONS—List Parameter Options

Determines whether or not to list execution time parameters. If LISTOPTIONS=Y is specified, all default parameters and parameters overridden by the TAOPT dataset are listed in the JESMSGLG. LISTOPTIONS=N suppresses this list. The delivered default is LISTOPTIONS=N.

LISTOPTIONS=X is a special setting for DKJOPTGN (DKJT1134). It is the equivalent of LISTOPTIONS=N if the TAOPT DD was not present in the jobstream, and the equivalent of LISTOPTIONS=Y if the TAOPT DD is present. LISTOPTIONS=X applies to DKJOPTGN (DKJT1134) only; it does not apply to TAOPT.

This option applies to this region's users, regardless of which TSR they access.

This option can be specified for all IBM Z Table Accelerator interfaces.

LOCKTIMERC—Lock Timer Wait Value

LOCKTIMERC=nnnnnn specifies the number of seconds (default 0) that IBM Z Table Accelerator should wait for a lock. When the LOCKTIMERC interval has passed, RC=71 is returned or the call abends depending on the status switch setting. A value of LOCKTIMERC=0 specifies that the process will never time out.

The lock controlled by LOCKTIMERC is used internally by IBM Z Table Accelerator to maintain table integrity in the TSR. It is unrelated to the table ENQUEUE that occurs when a table is opened for write (OW).

This option applies to this region's users, regardless of which TSR they access.

This option can be specified for all IBM Z Table Accelerator interfaces except the VTS interface.

LOCKTIMEWTO—Lock Timer Message Wait Value

LOCKTIMEWTO=nnnnnn specifies the number of seconds (default 30) to wait before issuing messages that the process is waiting for a lock. A value of LOCKTIMEWTO=0 specifies that no warning message will be issued.

The lock controlled by LOCKTIMEWTO is used internally by IBM Z Table Accelerator to maintain table integrity in the TSR. It is unrelated to the table ENQUEUE that occurs when a table is opened for write (OW).

When applicable, messages are generated to report that some processing has been blocked.

This option applies to this region's users, regardless of which TSR they access.

This option can be specified for all IBM Z Table Accelerator interfaces except the VTS interface.

MAXNMTAB—Maximum Number of Tables

Determines the number of tables that can be opened in a given local TSR or VTS-TSR. (With the possibility of creating a TSR of up to 2G in size, there can be a great variation in how many tables will actually be in a TSR at any time.) If the value of the MAXNMTAB is not set by the user (or is set to zero) then a default value is calculated.

If an attempt is made to open more tables than is indicated by MAXNMTAB, an error code of 20 will be given: The maximum number of tables has been exceeded. Check MAXNMTAB.

This option applies to the TSR created by this region/job and all users accessing it.

This option can be specified for all IBM Z Table Accelerator interfaces.

User sets the value of MAXNMTAB

To optimize the memory required for system overhead, the user may choose to set the value of MAXNMTAB rather than use the default value. MAXNMTAB can be set to any integer, zero or greater. However, if MAXNMTAB is set to a value greater than the default for the given TSR size, the default is used and the user is notified with a warning message.

After the user sets the value of MAXNMTAB, memory is allocated and an index is created that contains enough entries to support the maximum number of tables indicated. Additional memory is allocated as tables are opened. This memory is reused as tables are closed and new tables opened. This approach keeps the system overhead in line with the number of tables that are opened.

The calculation of the default value of MAXNMTAB

If the user does not set the value MAXNMTAB or the value is zero, the IBM Z Table Accelerator software calculates the default value based on the size of the TSR. Each table in the TSR occupies a minimum of 4K pages. The maximum possible number of tables is 51,455, for a 2G TSR; the default, when unspecified is one table for every 4K page in the TSR.

MTRETAIN—Retain Rows and Index Areas

MTRETAIN=Y|N determines whether the allocated rows and index areas are to be retained when an MT command is issued or whether they are to be reduced to the original allocation before table expansion. The default is "N", which is current processing—not retained.

Also see “ZEROROWS—Zero Data Table Rows on De-allocation” on page 31.

This option applies to the TSR created by this region/job and all users accessing it.

This option can be specified for all IBM Z Table Accelerator interfaces.

MULTITASKING—Multitasking

Specifies whether this region will allow multiple subtasks to access IBM Z Table Accelerator.

This option can only be specified for the IBM Z Table Accelerator batch interface. All other environments are multitasking in their nature, and will not accept this option.

The MULTITASKING option can be set to Y to allow multiple subtasks within a region to access IBM Z Table Accelerator concurrently. Each subtask may have multiple concurrent subtasks of its own, with no limit to the level of multitasking. The delivered default is N (multitasking not enabled).

This option applies to this region's users, regardless of which TSR they access.

Note:

1. If a batch application attempts to issue a IBM Z Table Accelerator command from a second TCB with Multitasking=N, it will abend with an S0C3 and display message: "DKJ00496E Multitasking requires TAOPT Multitasking=Y be set".
2. Please call Technical Support for assistance when writing applications that access IBM Z Table Accelerator in a multitasking environment.

OVRRIDES—Allow Changes to Status Switches

It is recommended that you do not allow this option to be changed using TAOPT. It should only be changed using DKJOPTGN.

The Status Switches may be changed by the application with the use of the Change Status (CS) command. Overriding individual Status Switches may be inhibited by the settings of the override controls. Overrides are specified as a string of Y’s and N’s; for example: YYYNYNNN. A value in any of the eight positions represents a change for that override. A * or blank in any of the eight positions represents no change for that override.

The defaults for these values will depend upon the interface. See the Implementation Guide for further details.

Table 1. Overrides default values

Position	**Description**	**Override setting**	**Meaning**
1	Allow Change To Abend On Errors	Y	The application is allowed to set the ABEND status switch.
		N	The application is not allowed to set the ABEND status switch.
2	Allow Change To Wait For Enqueued Tables	Y	The application is allowed to set the WAIT status switch.
		N	The application is not allowed to set the WAIT status switch.
3	Allow Change To Return Empty Rows In Hash Tables	Y	The application is allowed to set the HASH-EMPTIES-RETURNED status switch.
		N	The application is not allowed to set the HASH-EMPTIES-RETURNED status switch.
4	Allow Change To Permit Implicit Open Of Tables	Y	The application is allowed to set the IMPLICIT OPEN status switch.
		N	The application is not allowed to set the IMPLICIT OPEN status switch.
5	Allow Change To Trace IBM Z Table Accelerator Commands	Y	The application is allowed to set the IBM Z Table Accelerator TRACE status switch.
		N	The application is not allowed to set the IBM Z Table Accelerator TRACE status switch.
6	Reserved for future use; default set to Y.		
7	Reserved for future use; default set to Y.		
8	Reserved for future use; default set to Y.		

The values set by this option apply to this region's users, regardless of which TSR they access.

This option can be specified for all IBM Z Table Accelerator interfaces.

RACF_LIBACCESS—Perform Checks on Protected Libraries

Specifies whether IBM Z Table Accelerator should check if a IBM Z Table Accelerator library is SAF interface protected. Examples of products which implement SAF interface protection are RACF, ACF2 and Top Secret.

This option is only available in the IBM Z Table Accelerator Batch and IMS interfaces.

The following is a description of each parameter value:

- Y indicates that IBM Z Table Accelerator will check whether a library is protected. Any update attempts without update access to the library will fail with error code 61-12.

- N indicates that IBM Z Table Accelerator will not check whether a library is protected. Any update attempts without update access to the library will fail with a S913-38 for IBM Z Table Accelerator BDAM libraries and a U301 for IBM Z Table Accelerator VSAM libraries.

The delivered default for this parameter is N.

STROBE—Strobe Interval

Specifies a numeric value, from 0 to 2,147,483,647(2G – 1), to control the interval after which IBM Z Table Accelerator strobe reporting is triggered. The delivered default is 0.

For the VTS environment, this option is used in conjunction with STROBEMETHOD. The numeric value specified either represents a call count (STROBEMETHOD=0, 1 or 2) or a minute interval (STROBEMETHOD=3). See "STROBEMETHOD—Strobe Method" on page 27 for more details.

For all other environments, the numeric value represents a call count.

This option can be specified for all IBM Z Table Accelerator interfaces.

For the batch, IMS and VTS interfaces, a final strobe report will always be produced if the TATSRPT DD statement is present (even if STROBE is set to 0). In order to suppress strobe reporting, the TATSRPT DD statement must be removed from the job. For STROBE=0, no intermediate strobe report is produced.

TATSRPT is not relevant in the CICS environment. CICS strobe statistics are written to a CICS journal file (DFHJ99) from which the strobe report is generated via a IBM Z Table Accelerator utility (DKJTSTRB). No final strobe is written to the CICS journal if STROBE is set to 0 or the STROBE is set at a value which causes no strobes to be written before step end. This means that for these cases, there will be no STROBE report generated.

For more information on CICS strobe reporting, please refer to the *IBM Z Table Accelerator Administration Guide*.

This option applies to the TSR created by this region/job and all users accessing it.

Note: The numeric value must be specified without commas.

STROBEMETHOD—Strobe Method

Specifies when IBM Z Table Accelerator strobe reporting is triggered for the VTS-TSR.

This option applies only to the VTS environment.

Values for this parameter can be 0 and 3. The delivered default is 0.

The following is a description of each parameter value:

- 0 indicates that the traditional strobe reporting method, by number of calls, using the STROBE=n parameter, will occur
- 3 indicates that strobe reporting will be triggered by time interval instead of call count. STROBE=n will indicate the number of minutes, n, after which a strobe is to be taken.

If STROBE=n is not specified with STROBEMETHOD, the default of STROBE=0 is used and only 1 report will be produced at IBM Z Table Accelerator termination.

STROBEMETHOD can be specified in the TAOPT DD statement or modified in DKJOPTGN source, reassembled and relinked into DKJVBASE for the VTS Agent job.

For STROBEMETHOD of 0, IBM Z Table Accelerator processing is prevented from waiting for the strobe report. If the number of calls in the STROBE parameter is set to a small value and IBM Z Table Accelerator calls are made more frequently than strobe reports can be produced, the frequency of reports may not match the STROBE value.

SUPPRESS_DUMPS—Suppress IBM Z Table Accelerator Messages and Dumps

Specifies whether to suppress IBM Z Table Accelerator messages and dumps when an abend occurs.

This option applies only to the CICS environment.

Values for this parameter can be NONE or SYMPATHETIC. The delivered default is SUPPRESS_DUMPS=NONE.

The following is a description of each parameter value:

- NONE indicates that all IBM Z Table Accelerator messages and dumps will be produced.
- SYMPATHETIC indicates IBM Z Table Accelerator error messages and dumps (TADUMP and IBM Z Table Accelerator CICS dumps) will be suppressed if a transaction abends outside IBM Z Table Accelerator code (i.e., in the associated application code, or in some other non-IBM Z Table Accelerator code). Any abends occurring within IBM Z Table Accelerator code will still produce the IBM Z Table Accelerator error messages and dumps.

SWITCHES—Status Switches

The Status Switches are a series of one-byte codes for altering the operation of the Application Programming Interface used by your programs when requesting a IBM Z Table Accelerator service. Switches are specified as a string of Y's and N's; for example: YYYNYNNN. A value in any of the eight positions represents a change for that override. A * or blank in any of the eight positions represents no change for that override. For information about switch settings, see Table 2 on page 28.

The switches are stored in the parameter module for the interface. As each batch job or online task begins, the run-time values will be set according to the values in the parameter module but may be changed using the values in the TAOPT file. For more information, see Parameters in the MVS Batch section of the *IBM Z Table Accelerator Implementation Guide*.

The application also may change the values with the CS, the Change Status command (unless changing a particular switch is suppressed via the override controls).

The defaults for these values depend upon the interface and are described in the *IBM Z Table Accelerator Implementation Guide* in the appropriate chapter:

- Implementing IBM Z Table Accelerator z/OS Batch
- Implementing the CICS Interface
- Implementing the IMS TM Interface

Table 2. Switches default values

Position	Default	Switch setting	Meaning
1	Abend on Errors	Y	Abend processing is to be performed on IBM Z Table Accelerator errors 0001-0099 or 1001-1099. Errors 100 to 999 and errors greater than 1099 always abend.
		N	Abend processing is not to be performed on IBM Z Table Accelerator errors 0001-0099 or 1001-1099. User programs could handle these return codes.

Table 2. Switches default values (continued)

2	Wait for Enqueued Table	Y	IBM Z Table Accelerator is to wait for tables that are enqueued.
		N	IBM Z Table Accelerator is not to wait for such enqueued tables. In this case IBM Z Table Accelerator will terminate with a user 0072 abend if abend processing is enabled, or, will return an error code of 0072: Table unavailable; no wait in the command area if abend processing has been disabled.
		Note: Waiting in an online environment is normally discouraged. In CICS and IMS, the default is set to N	
3	Return Empty Rows in Hash Tables	Y	IBM Z Table Accelerator is to return empty rows for hash tables
		N	IBM Z Table Accelerator is to suppress the return of empty rows for hash tables
4	Allow Implicit Open Of Tables	Y	IBM Z Table Accelerator is to automatically open tables for read on first access
		N	IBM Z Table Accelerator is to suppress automatic opens; an explicit open command, OR, OW, or IA must be issued to open a table
		Note: We recommend setting Allow Implicit Open of Tables to N, especially in multi-user environments. This gives better control over which tables are opened.	
5	Trace IBM Z Table Accelerator Commands	Y	IBM Z Table Accelerator automatically records the last ten commands executed per thread. This is done for diagnostic purposes.
		N	IBM Z Table Accelerator does not trace commands
6	Reserved for future use; specify as N.		
7	Reserved for future use; specify as N.		
8	Reserved for future use; specify as N.		

The values set by this option apply to this region's users, regardless of which TSR they access.

This option can be specified for all except the VTS Interface.

TABLEWAITRC—Table Open Enqueue Wait Time

Specify a value to indicate the number of seconds that a user will wait to obtain the MVS enqueue to open a table for read or write before timeout. If the enqueue is not obtained before the timeout, IBM Z Table Accelerator will return code 72, or will abend, if the *Abend on Errors* switch is set to Y.

The delivered default is 0 (wait forever).

The Wait for Enqueued Table switch (see Table 2 on page 28) must be set to Y for this parameter to have any effect.

This option applies to this region's users, regardless of which TSR they access.

This option can be specified for all except the VTS Interface.

TABLEWAITWTO—Table Open Enqueue Report Time

Specify a value to indicate the elapsed time before a message will be generated to report that the enqueue has not yet been received.

The delivered default is 30 (0=no messages).

The Wait for Enqueued Table switch (see Table 2 on page 28) must be set to Y for this parameter to have any effect.

This option applies to this region's users, regardless of which TSR they access.

This option can be specified for all except the VTS Interface.

TSR_ALGORITHM—Optimize TSR Usage

TSR_Algorithm=P|D allows the TSR Space Manager to determine if TSR space usage should be maximized or performance should be optimized when TSR space is allocated and deallocated for table usage. The TSR space manager is the IBM Z Table Accelerator component responsible for tracking TSR space usage and for allocating and deallocating TSR space when tables are opened, closed, expanded or reduced in size.

The following is a description of each parameter value:

- P (Performance) indicates that the TSR will be optimized for performance. Space will be assigned to tables within the TSR so as to minimize CPU usage, which may result in a less than optimum use of space.
- D (Default) indicates that there will be no optimization, and there will be no messages regarding optimization. However, messages will be provided regarding current percentage of TSR capacity.

The delivered default is P.

This option applies to the TSR created by this region/job and all users accessing it.

This option can be specified for all IBM Z Table Accelerator Interfaces.

TSR_WARNING_FREQ—Frequency of TSR Allocation Warnings

Specify a value to indicate the frequency of reports generated when the TSR allocation percentage (as defined by TSR_WARNING_PCT) is exceeded. 0 results in a message for every allocation over the specified percentage; 30 results in a wait of at least 30 seconds before repeating the message; 999 results in a wait of just over 16 minutes before repeating the message.

The delivered default is one warning every second (001).

This option applies to this region's users, regardless of which TSR they access. (Different regions accessing a VTS-TSR can set their own thresholds.)

This option can be specified for all IBM Z Table Accelerator Interfaces.

TSR_WARNING_PCT—Percentage of TSR Allocation for Warning Activation

Specify a value to indicate the percentage of TSR allocation allowed before a message will be generated to report that the allocation percentage has been exceeded. The message will be repeated on every allocation of this percentage, subject to the TSR_WARNING_FREQ parameter.

The delivered default is 85 (85%).

This option applies to this region's users, regardless of which TSR they access. (Different regions accessing a VTS-TSR can set their own thresholds.)

This option can be specified for all IBM Z Table Accelerator Interfaces.

TSRSIZE—tableSPACE Region Size

The TSRSIZE parameter is an integer representing the amount of storage to be used for the TABLE SPACE REGION (TSR). IBM Z Table Accelerator uses Data Spaces for all TSRs, whether local or VTS. This ensures that tableSpace memory requirements do not affect other memory requirements in the region. The format for TSRSIZE is:

- nnnnnnnn = bytes
- nnnnnnnK = kilobytes
- nnnnM = megabytes
- nG = gigabytes

Value recommendations

The delivered default value for the TSRSIZE is 10M for all interfaces. The current minimum size of a TSR is 28 KB (but this may change in subsequent maintenance releases). The maximum size of a TSR is 2G (or 2048M or 2097152K). TSRSIZE only accepts up to eight digits, thus you cannot specify 2G as TSRSIZE=2147483648.

Note: If you specify a value of 0, the minimum size will be substituted; if you specify a value of between 1 and the minimum, an error code will be returned.

Other considerations

You must ensure you are allocating enough space for all tables open simultaneously in the region. You can determine the TSR size in one of two ways:

1. use the LT command in the batch program DKJTDRV, CICS transaction TADR,
2. browse strobe reports produced by your job runs.

Failure to set your TSR size parameter appropriately can produce the following error:

Error 92	There is insufficient tableSPACE region available. Increase TSR size.

Note: There is a possibility that your system programmers have established limits on the maximum size or the maximum number of Data Spaces. All local TSRs are allocated as SCOPE=SINGLE Data Spaces; VTS-TSRs are allocated as SCOPE=ALL Data Spaces.

This option applies to the TSR created by this region/job and all users accessing it.

This option can be specified for all Interfaces.

VTSNAME—Specifying the Name of a VTS-TSR

The VTSNAME option specifies the name to be assigned to the VTS-TSR when it is initialized by the VTS Agent. The name is used by other regions to access the shared TSR.

Notes:

1. VTSNAME can be 1 to 8 characters. The naming restrictions are identical to those for DDNAMEs in IBM JCL.

This option applies to the TSR created by this region/job and all users accessing it.

ZEROROWS—Zero Data Table Rows on De-allocation

Related to the MTRETAIN parameter (see "MTRETAIN—Retain Rows and Index Areas" on page 25)—applies only when MTRETAIN = N.

ZEROROWS=Y|N determines whether the data rows area should be zeroed when it is deallocated. The default is Y, which is current processing. Note that the index area is never zeroed, except for hash indexes.

This option applies to the TSR created by this region/job and all users accessing it.

This option can be specified for all IBM Z Table Accelerator interfaces.

Appendix B. TAOPT dataset coding

The TAOPT dataset can be a sequential file, a member of a data set, or, for CICS, a VSAM dataset. The TAOPT dataset can be specified for all interfaces, including VTS. The dataset must contain fixed-length 80-byte records.

The data in TAOPT uses the same parameter names and values as are coded on the DKJOPTGN macro for the defaults, with the exception of LIB-LIST. TAOPT uses LIBnn to specify IBM Z Table Accelerator libraries to update the IBM Z Table Accelerator Library List.

Each parameter is entered on a single line in the dataset. The parameter may begin in any column. A line beginning with an asterisk (*) denotes a comment. Comments may also be added after the parameter value. A semicolon may be used to indicate line end. Comments may follow the semi-colon.

Although each region may have defined its own TAOPT dataset, all regions can share a sequential DASD dataset, and CICS regions can share a VSAM TAOPT dataset.

A sample TAOPT dataset for a batch region follows:

```
//TAOPT DD *
* A leading asterisk denotes a comment
  ListOptions=Y
  TSRSIZE = 12M
  MAXNMTAB=500
  LIB01 = TESTLIB; Testlib is searched first
  LIB02 = MAINLIB; Mainlib is searched second
/*
```

Figure 6. Sample TAOPT

Note:

1. ListOptions=Y is handy for diagnostic purposes; TESTLIB is first for batch testing.
2. With the exception of the LISTOPTIONS parameter, parameters must appear only once in the TAOPT file. The form KEYWORD=* indicates that the site default is to be used. The form KEYWORD=0 indicates that the default value of a parameter that takes a character string be nullified.

Appendix C. IBM Z Table Accelerator LPA-eligible programs

Table 3 on page 35 lists modules that are recommended to be stored in LPA (Link Pack Area) for the Batch and IMS/TM environments.

Table 3. Recommended LPA-eligible modules for Batch and IMS TM environments

DKJTCALL	DKJTNAME
DKJTNUCL	DKJTROTB

Note: The following modules cannot reside in LPA: DKJBBASE, DKJIBASE, DKJVBASE.

Notices

Trademarks

SC28-3232-00