

**Technical Paper**

# **Graceful Switchover and Switchback**

## **Oracle Standby Database**

Versions 7.3, 8.0, 8.1  
Revised November 11, 1999

**Lawrence To**

**Center of Expertise  
Oracle Support Services  
Oracle Corporation**

## Graceful Switchover and Switchback

<b>INTRODUCTION</b>	<b>5</b>
<b>TERMINOLOGY</b>	<b>6</b>
<b>DESCRIPTION OF STANDBY DATABASES</b>	<b>9</b>
<b>ADVANTAGES AND DISADVANTAGES</b>	<b>9</b>
<b>PREREQUISITES OF GRACEFUL SWITCHOVER OR SWITCHBACK</b>	<b>11</b>
<b>EVENTS THAT PREVENT GRACEFUL SWITCHOVER OR SWITCHBACK</b>	<b>12</b>
<b>GRACEFUL SWITCHOVER STEPS</b>	<b>12</b>
<b>GRACEFUL SWITCHBACK STEPS</b>	<b>21</b>
<b>GRACEFUL SWITCHOVER AND SWITCHBACK EXPLAINED</b>	<b>22</b>
Recovering and Opening a Standby Database	23
Converting a Standby Database into a Production Database	23
Converting Production Database into a Standby Database	24
<b>BEST PRACTICES</b>	<b>24</b>
<b>APPENDIX A - INIT.ORA FILES</b>	<b>26</b>
standby init.ora file	26
production Init.ora file	27
<b>APPENDIX B - QUERIES AND CHECKS</b>	<b>28</b>
Queries to Validate Production Database	28
Finding Database Resetlogs Version	30
<b>APPENDIX C - CREATE CONTROLFILE SCRIPT</b>	<b>31</b>

<b>APPENDIX C - POST OPEN SCRIPT</b>	<b>32</b>
<b>APPENDIX D - CONSIDERATIONS FOR OPS</b>	<b>34</b>
<b>ACKNOWLEDGMENTS</b>	<b>36</b>
<b>RELEVANT BUGS</b>	<b>37</b>

Copyright © Oracle Corporation 1998. All rights reserved. Printed in the USA.

Author: Lawrence To

Oracle is a registered trademark of Oracle Corporation. Oracle7, Oracle8 and Oracle8i are trademarks of Oracle Corporation. All other products or company names are used for identification purposes only, and may be trademarks of their respective owners.

# GRACEFUL SWITCHOVER AND SWITCHBACK

LAWRENCE TO

*Oracle Corporation is not documenting graceful switchover and switchback between production and standby databases until Release 8.1. However Oracle is officially supporting the graceful switchover steps if the customer is using a supported version of standby database, tested and validated all steps, and followed the recommended steps and best practices. A user error can seriously harm your standby or production database; therefore, a thorough understanding of graceful switchover and switchback is mandatory.*

*The Centers of Expertise (COE) within Oracle Support Services (OSS) and Oracle's mission-critical customers understand the advantages of having a graceful switchover and switchback mechanism; therefore, we at the COE would like to share our best practices in implementing safe, customized graceful switchover and switchback. At the conclusion of this paper, you will have:*

- *an understanding of the advantages of graceful switchover and switchback*
- *a list of steps to safely implement switchover and switchback*
- *an understanding why these steps work and are safe*
- *a method to validate the integrity of production and standby databases*

*Audience: Senior DBAs or system administrators with a thorough understanding of Oracle standby database implementation and Oracle recovery mechanisms.*

## Introduction

This paper educates sophisticated DBAs and system administrators on how to enhance their current standby database environment. It provides checks and balances and explanations to build graceful switchover and switchback features. With graceful switchover and switchback best practices, customers will be able to

- avoid opening the targeted database with the resetlogs operation **under certain conditions**,
- reduce vulnerability of primary database due to loss of disaster recovery site, and
- simplify system administration without jeopardizing the integrity of the database.

Graceful switchover or switchback is particularly useful when switching back from a standby database to the primary database. After a disaster on the primary site, disaster recovery occurs and the standby database is opened. After the primary site is repaired and the primary database replaced, customers prefer a graceful switchback from standby to primary rather than invalidating a database again. More details will be described in the body of the paper.

*Important Note: These best practices and steps must be rigorously tested and understood because a mistake can result in the loss of the production or standby databases. Contacting the OSS or COE is highly recommended if there are issues in setting up*

## Graceful Switchover and Switchback

*graceful standby database implementations that differ from what is described in this paper. Customers must avoid using their own version of standby database switchover and switchback to safeguard the integrity of their databases.*

### Terminology

Unfortunately, the description of standby databases includes a spectrum of new and sometimes vague terms. To minimize the confusion, we like to define some of most common terms used in this paper.

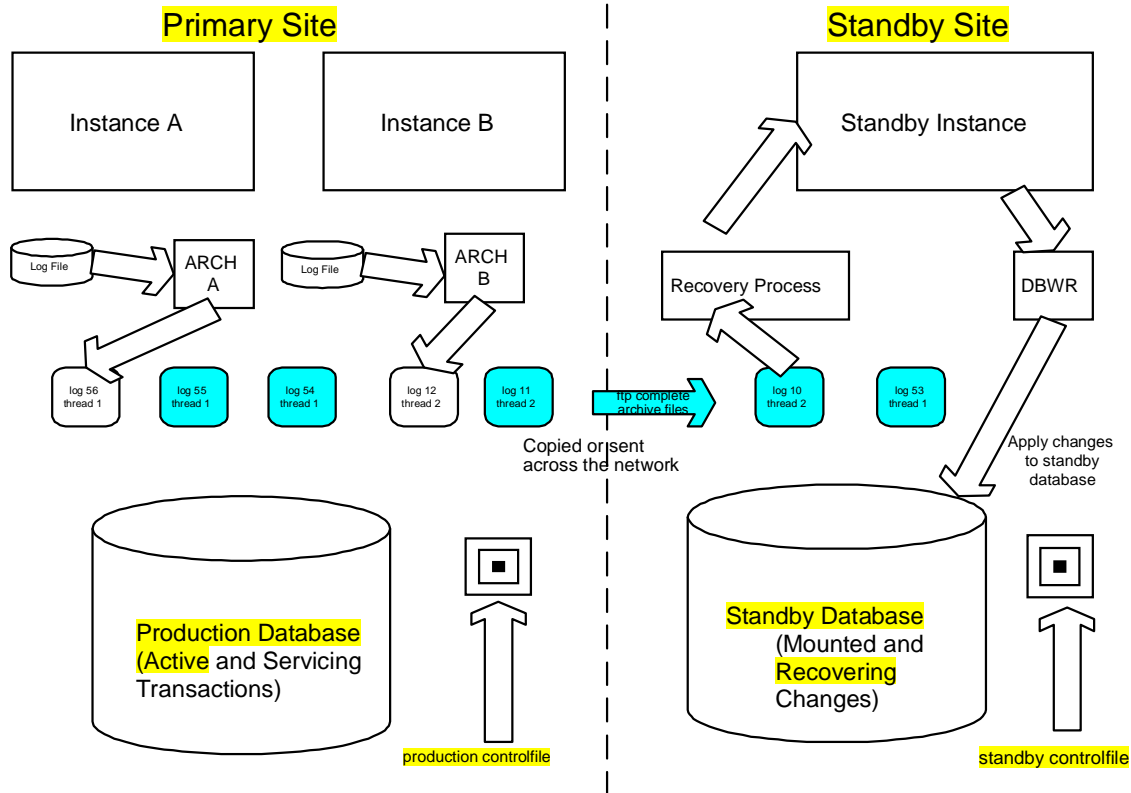
- Production/Active database - the database servicing your business application. This database is active and open.
- Standby/Recovering database- the standby database which is in mounted and in recovery mode.
- Primary site - the site is the **initial** source for your production database and system resources that service your business applications. The location and place are constants.
- Standby site- the site that initially contains the machine(s) for standby database. The location and place are constants.
- Source database- we are switching from this database. Previously the source database was our production database but after the switch, it becomes the standby database.
- Target database- we are switching to this database. Previously the target database was our standby database but after the switch, it becomes the production database.
- Primary controlfile- controlfile of the production database.
- Standby controlfile- controlfile of the standby database. It contains a flag indicating that this is a standby database that only allows certain standby operations such as mount and recovery until it is activated. Once it is activated, it becomes a production database.

## Graceful Switchover and Switchback

Figures 1 and 2 help illustrate the previous terms.

Figure 1, *Standby Database*, depicts a typical environment that contains a primary and standby site. The primary site contains the production database servicing your business. All production databases must be in archive log mode. As archive redo logs are produced, they are sent to the standby site, which eventually applies the logs to the standby database through Oracle's recovery mechanism. The standby database is not open; instead, it is in a mounted state being fed production database changes by the archive (redo) stream. The vertical dotted line in the middle of the figure indicates that the two sites are physically separate and the transfer of files occurs through the network.

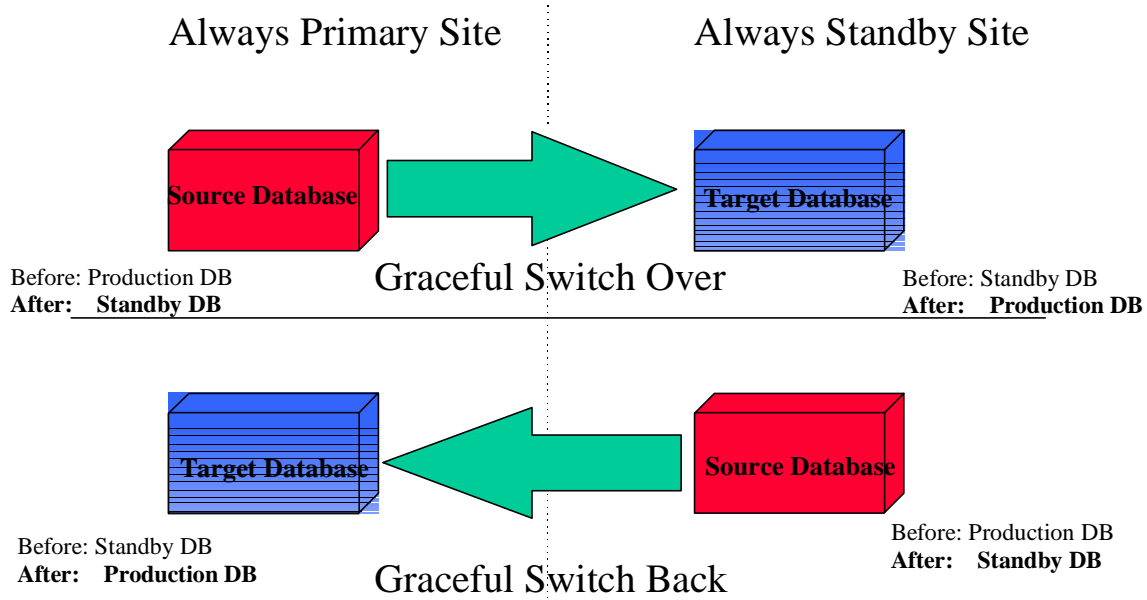
Figure 1: Standby Database



## Graceful Switchover and Switchback

Figure 2, *Graceful Switchover and Switchback*, illustrates the outcome of the source and target databases after a graceful switchover and switchback. Notice that the Primary and Standby sites are always constant regardless where the production and standby databases reside. In fact, the fundamental goal of these procedures is to seamlessly and painlessly switch the roles of the standby and production database. Finally, the dotted line in the middle indicates that source and target databases are physically separate.

Figure 2: Graceful Switchover and Switchback



## Description of Standby Databases

This section describes standby databases and how customers implement switchover and switchback today.

Standby databases are critical components in achieving fast and safe disaster recovery. They provide a redundant copy of the production database that is physically stored at a remote site. The standby database is updated by the production database's archive logs which keeps a record of physical database changes except for "unrecoverable" or "unlogged" changes. It is the most common disaster recovery scheme for Oracle mission critical customers. When standby databases are implemented with care, they offer an efficient disaster recovery and failover solution. Standby databases augment other available recovery strategies such as failover with Oracle Parallel Server or Hardware Fail Over systems, on host recovery schemes such as triple mirroring, Oracle replication, custom replication, EMC SRDF failover solutions or the archaic, but necessary tape restore and recovery.

Standby databases provide the following advantages:

- failover to a remote location upon disaster (loss of the primary site),
- faster recovery and lower MTTR (mean time to recover),
- potentially no data loss if online redo logs are applied,
- no or little performance degradation on the primary site (unlike Oracle replication and most OPS fail over environments that incur a performance overhead).
- feature is inherent to Oracle's database product (lower cost alternative, as the Standby instances are only a fraction of the primary's cost)

Today, customers can switch over or switch back between production and standby databases. However, they pay an enormous price by invalidating previous backups and leaving a large window of vulnerability. The window of vulnerability is dependent on the time it takes to recreate your standby database which can be as much as 4 to 10 hours for large databases. Unfortunately, the only supported mechanism requires an *open resetlogs* operation of the targeted database. This *resetlogs* operation has an enormous impact on both target and source databases. The *resetlogs* operation invalidates all previous backups of your target database including your source database by making the new production database incompatible with previous backups. Subsequently, new backups and a new standby database need to be created. This equates to a large window of vulnerability (not having a standby database), more system administrator work, and loss of considerable amounts of money and time if an outage hits your production database while you are still rebuilding your standby database.

Some customers try to work around this inconvenience by inventing creative ways to bypass the *resetlogs* operation such as switching primary and standby controlfiles. However, these innovators or hackers usually find themselves faced with an unexplained Oracle corruption and loss of the production database.

## Advantages and Disadvantages

The main advantage of a graceful switchover or switchback is that it avoids the *resetlogs* operation. By avoiding the *resetlogs* operation, the source database can resume its role as the standby database almost immediately. A more detailed description of the advantages and disadvantages are described in the table 1.

Graceful Switchover and Switchback

Table 1: Advantages and Disadvantages

Advantages	Disadvantages	Perceived Value Gain or Value Loss	Measured Value Gain or Value Loss
<p>Reduced MTTR in case new target/production database fails</p> <p>No need to recreate source/standby database.</p> <p>Save hours of backup time (e.g. estimated 3-9 hours)</p> <p>Does not invalidate previous backups.</p> <p>Reduced system administration</p>		<p>High perceived value.</p> <p>Dependent on cost of downtime.</p> <p>High perceived value.</p> <p>Reduced vulnerability. Reduce MTTR. Reduce administration.</p> <p>High perceived value.</p> <p>Reduced vulnerability. Reduce MTTR. Reduce administration.</p> <p>Medium Gain</p>	<p>High measured value.</p> <p>Dependent on cost of downtime.</p> <p>High measured value.</p> <p>Dependent on cost of downtime.</p> <p>High measured value.</p> <p>Dependent on cost of downtime.</p> <p>Medium Gain</p>
	<p>User error can result in loss of production database or unexplained Oracle corruption.</p> <p>Slower switch over and switch back times compared to disaster recovery switch over.</p> <p>Estimated 6-20 minutes overhead. Please refer to chart, Graceful Switchover Steps.</p> <p>Standby database is unavailable for querying or OLTP. No scalability gain for production database.</p> <p>Lack of protection for user error or database corruption.</p>	<p>High perceived value loss. Avoided by rehearsal and practice.</p> <p>Medium perceived value loss. Gain in most cases outweighs loss.</p> <p>High Value Loss</p> <p>Offset by reduced MTTR and higher availability.</p> <p>8.1 provides READ ONLY functionality.</p> <p>Medium Value Loss</p> <p>Offset by monitoring, error checking and sound operational practices.</p>	<p>High Value Loss</p> <p>Avoided by rehearsal and practice.</p> <p>Low Value Loss</p> <p>Dependent on cost of downtime.</p> <p>Low Value Loss</p> <p>Low Value Loss</p>

## Prerequisites of Graceful Switchover or Switchback

The prerequisites of initiating graceful switchover and switchback are the following:

1. production database is shutdown normal or shutdown immediate or instance down,
2. there is no loss of any archive logs that haven't been (yet) applied to the standby database,
3. all archives are applied to the standby database,
4. standby database is shutdown normal or shutdown immediate,
5. **source** database's online redo logs are available and intact,
6. **target** database is still intact (data files, controlfiles are present) and has the same *resetlogs* version as the source database, and
7. loss of unrecoverable transactions (e.g. direct loader unrecoverable) is acceptable
8. all steps have been validated and tested in a test environment first
9. experienced DBAs with in-depth knowledge of database and standby recovery are managing the environment

Graceful switchover and switchback will not work when either production database's online redo logs or standby database is lost or inaccessible. The source database must be shut down successfully (*shutdown normal* or *shutdown immediate*) or at the very least, the source instance needs to be unavailable. Subsequently, the target database must apply all of the source database's archive redo logs (not online redo logs<sup>1</sup>) and then must be shut down successfully. Now target to source role reversal is accomplished by

- recreating the target controlfile<sup>2</sup>, and
- recovering the database and applying the source's online redo logs<sup>3</sup>.

Copying online redo log files must occur while databases are shut down. The online redo logs can also be geo-mirrored (e.g. remote mirroring, EMC SRDF synchronous mode) between the standby and primary sites to ensure accessibility of the online redo logs in case of a primary site failure. The **target** database now can be mounted and opened as the **production database**. Afterwards, the production database creates a separate standby controlfile (e.g., *alter database create standby controlfile as 'filename'*). This standby controlfile is copied to source database and used to complete the source to target role reversal. The **source** database will then be mounted with the standby controlfile as a **standby database** and begin standby recovery. Criteria 1-5 are upheld.

Criteria 6 implies that both target and source databases **must** have the same *resetlogs version*. The *resetlogs version* determines if the current database is compatible with previous backups. In essence, the target database is a backup or subset of the source database. In most cases, both databases will have a *resetlogs version* of 0 which

---

<sup>1</sup> It is never recommended to apply online redo logs directly since it may inadvertently stop recovery when recovering the current log. If you apply the online redo logs directly in a graceful case, the graceful switchover will fail.

<sup>2</sup> Previous revisions of the graceful switchover and switchback paper stressed that the production controlfile and the online redo logs need to be copied to the standby database; however we have found problems with controlfile transaction count mismatches between data file headers and controlfiles (bug 1034927). Oracle now recommends creating a "create controlfile with noresetlogs option" script from the production database and copying the script to the standby database before switchover occurs. The *alter database backup controlfile to trace noresetlogs* command must be executed on the production database instead of the standby database (bug 1034871).

<sup>3</sup> If the production database is not Oracle Parallel Server, only the online redo logs are required for startup after recreating the controlfile; however OPS environments may require additional work. Please refer to "Considerations for OPS" section in Appendix D.

implies that neither have ever executed a *resetlogs operation*. The *resetlogs version* is only updated after a *resetlogs operation*.

The seventh criterion is subtle but very important. Standby databases never recover from “unrecoverable” transactions or operations such as *direct loader unrecoverable* or *create table as select unrecoverable*. This is a well-known restriction and limitation of standby databases. When doing a role reversal, the standby database’s data files do not have changes from “unrecoverable” transactions.

### Events that Prevent Graceful Switchover or Switchback

It is sometimes easier to understand when a graceful switchover and switchback are not possible. **Graceful switchover and switchback are never possible when the production database's online redo logs are not accessible.**

A graceful switchover or switchback is also **not** possible whenever the production or standby databases execute one of the following:

- *alter database open resetlogs* or
- *alter database activate standby database* (which does an implicit *resetlogs operation*),

After a database undergoes a *resetlogs*, the database’s *resetlogs version* is incremented and redo beyond the incomplete recovery SCN is discarded. All previous backups including the source database cannot be resynched with the new target database (Exceptions are Read-Only and Offline normal files). The primary reason is recovering through a *resetlogs* is not supported.

Note: Refer to Appendix Section, Finding Database Resetlogs Version, for more information.

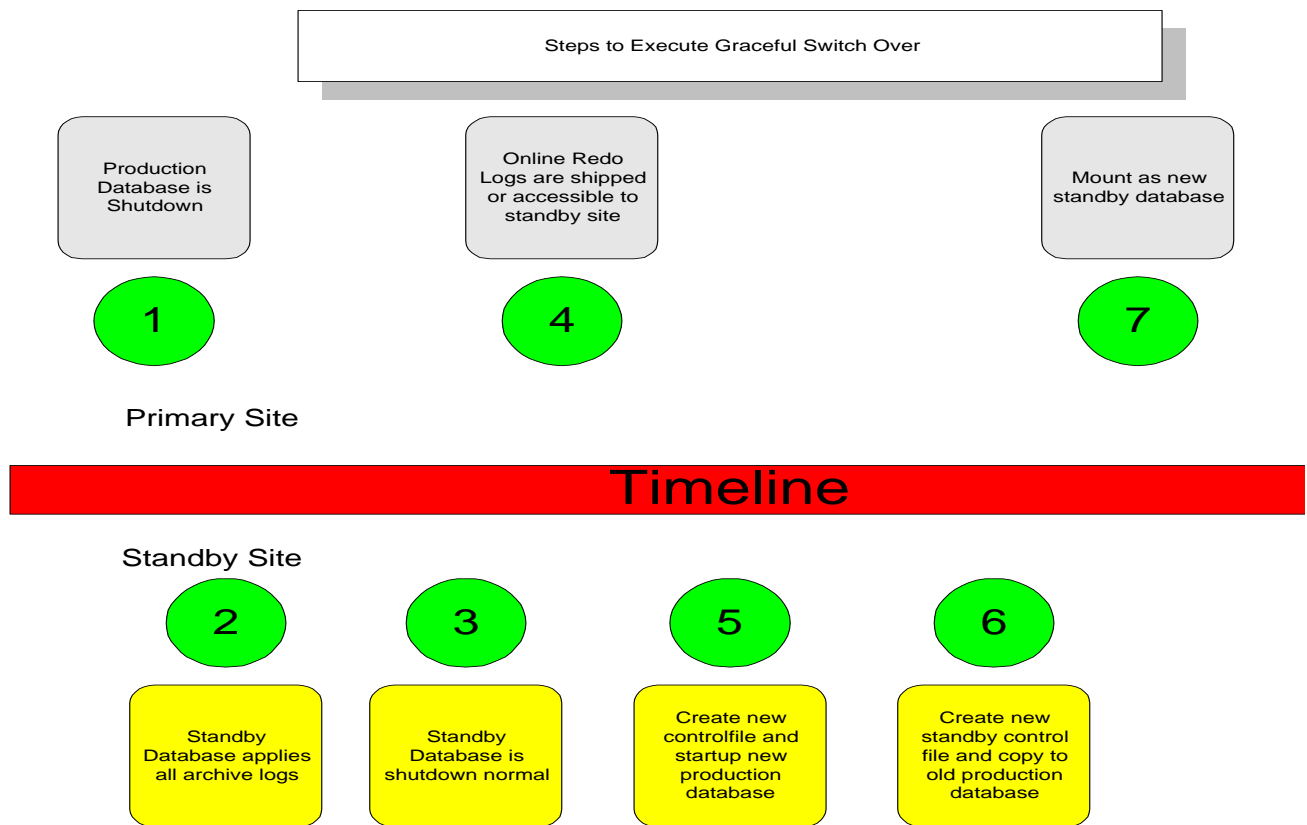
The only time either of those commands is required is when you are opening a database after one of the following recovery options:

- point in time recovery (*recover database until [cancel/change/time]*) or
- recovering with a backup controlfile (*recover database ... using backup controlfile*) or
- recovering a standby database (*recover standby database... or recover managed standby database*) or
- recovering a clone database for TSPITR (tablespace point in time recovery).

### Graceful Switchover Steps

The basic steps for graceful switchover are easy; however, the impact of an error can endanger the integrity of your production or standby databases. The following picture illustrates the 7 basic steps to complete a graceful switchover while table 2 provides more details.

Figure 3: Graceful Switchover Steps



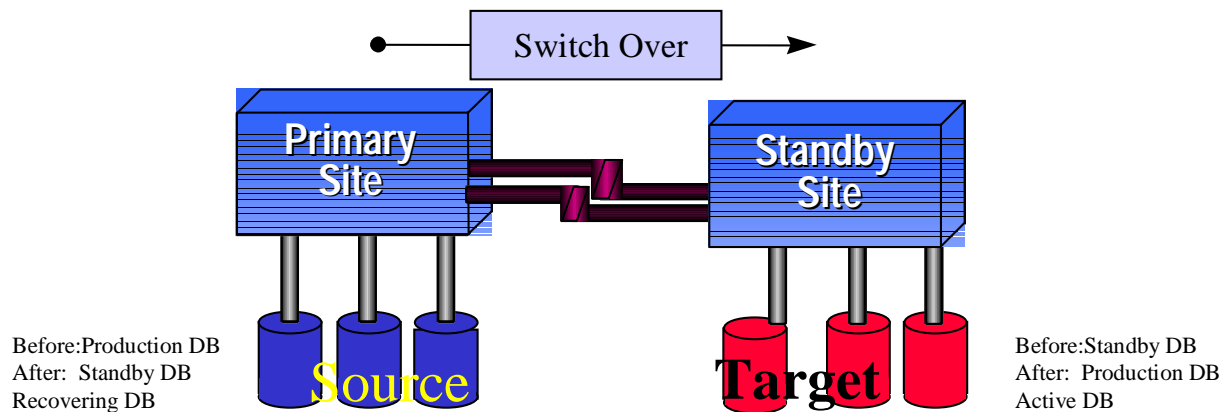
When the primary site requires scheduled maintenance, the production database can gracefully switch over to the standby database. Thereby, the graceful switchover technique may be useful for planned site repairs, hardware changes, operating system upgrades that does not require database upgrades or application changes. Any graceful switchover must be tested prior to executing in a production environment.

## Graceful Switchover and Switchback

Graceful switchover can also be leveraged for unscheduled outages if the primary site's online redo logs are accessible; however the risk of problems is higher since the integrity of the online redo logs is not guaranteed.

Figure 4, *Graceful Switchover*, describes the before and after states of the target and source databases. Notice that both databases are switching roles.

Figure 4: Graceful Switchover



At this stage, everyone must have a solid understanding of standby databases and the advantages of graceful switchover. We now attempt to list all the steps required for graceful switchover.

Prior to attempting the switchover, the source database contains the production database processing transactions and servicing clients. The standby database has the following status:

- database is mounted,
- database is in cancel-based recovery, and
- recovery was initiated by either the following commands: *recover standby database* or *recover managed standby database*.

The goal once again is to switch the roles of the production and standby databases.

Table 2, *Graceful Switchover Steps*, describes the graceful switchover steps.

**Table 2: Graceful Switchover Steps**

Identical numbers indicate that these steps can be executed concurrently.

Primary Site Production/Active Database and Source Database	Standby Site Standby/Recovering Database and Target Database
<p>Current Status: Production Database</p> <p>1. Discontinue client logons and complete last transactions. Clients log out.</p> <p>2. Archive the current log</p> <p>SVRMGR &gt; <i>alter system archive log current</i></p> <p>This command will archive all archive log groups from all enabled threads.</p> <p>2b. Create controlfile script and send to standby site.</p> <p>SVRMGR &gt; <i>alter database backup controlfile to trace noresetlogs</i></p> <p>The trace file that contains the script needs to be modified with the appropriate data file and log file path names. The new create controlfile script needs to be shipped to the standby site.</p> <p>We recommend creating a new controlfile script or maintaining an up to date controlfile script every time there is a database change such as adding a data file or data file status change. The script needs to be modified and prepared for the standby database. Please refer to Appendix C.</p> <p>3. Shutdown immediate or normal.</p> <p>SVRMGR &gt; shutdown immediate</p> <p>Verify that shutdown is completed successfully by looking at the alert.log. Verify that "Completed: ALTER DATABASE CLOSE NORMAL" is in the production's alert.log file.</p> <p>If the primary site crashed and is not accessible, these steps will still work if the online redo logs</p>	<p>Current Status: Standby database in recovery mode.</p> <p>1. Continue to apply archive logs.</p> <p>Setup for client failover to standby site.</p> <p>3. Copy the <i>controlfile</i> script from production to standby. Validate its accuracy.</p> <p>Refer to Appendix C.</p> <p>4. Apply last set of archives and Cancel Recovery. <sup>4</sup></p> <p>SVRMGR &gt; Cancel</p>

<sup>4</sup> Never apply online redo logs. When we apply the online logs as part of standby recovery, we determine that this is the end of all redo or end of thread (EOT) which forces a media recovery checkpoint. This leads to a checkpoint count mismatch which prevents the standby database from opening as the production database.

## Graceful Switchover and Switchback

are still intact and accessible.	
----------------------------------	--

Graceful Switchover and Switchback

Primary Site	Standby Site
Production/Active Database and Source DB	Standby/Recovering Database and Target DB
<p>4. Copy production's online redo logs from Primary to appropriate Standby site's location.</p> <p>Do not remove or delete these files from Primary Site.</p> <p>If the online redo logs are geo-mirrored using remote mirroring, they can then be utilized immediately.</p> <p>Ensure all logs have been sent to the standby.</p> <p>6. Execute scripts to reverse production and standby sites roles. Unfortunately, these scripts are customized and not provided by Oracle. Basically, you must have scripts for both production and standby roles located on each site.</p>	<p>5. Shutdown immediate or normal standby database.</p> <p>SVRMGR &gt; shutdown immediate</p> <p>Verify that shutdown is completed successfully by looking at the corresponding alert.log. This must be scripted and verified before proceeding.</p> <p>(i.e. Completed: ALTER DATABASE CLOSE NORMAL and ALTER DATABASE DISMOUNT messages can be found in the corresponding alert.log)</p> <p>6. Execute scripts to reverse production and standby sites roles. Unfortunately, these scripts are customized and not provided by Oracle. Basically, you must have scripts for both production and standby roles located on each site.</p> <p>7. Execute create controlfile script (e.g. CreateProductionCtl.sql - Appendix C)</p> <ul style="list-style-type: none"> <li>• <i>Startup nomount pfile=init.ora</i></li> <li>• <i>Create controlfile</i></li> <li>• Comment out <i>Recover database</i> command</li> <li>• Comment out <i>Alter database open</i> command</li> <li>• Comment out any additional commands after the <i>alter database open</i> command for later use. Refer to step 12.</li> </ul> <p>At the completion of the script, the database should be mounted.</p> <p>This init_production.ora is not the same as standby init.ora file. This step implies that there are separate production and standby init.ora and config.ora files. Each initialization file will point to the correct controlfiles and the appropriate parameters set. Separate startup scripts are also recommended.</p> <p>There may be special considerations for OPS.</p> <p>Refer to the Appendix A, C and D.</p>

Graceful Switchover and Switchback

Primary Site	Standby Site
<p>13. Receiving archives from new production database.</p> <p>16. Validate existence of all Oracle data files and standby controlfiles.</p>	<p>8. Validate the status and existence of all data files and log files.</p> <p>All data files required for production must be online. Data files from offline tablespaces and read only tablespaces will be offlined in the create controlfile script.</p> <p>9. <i>Recover database</i></p> <p>Recover command only recovers online data files. Please refer to Appendix D for OPS considerations.</p> <p>10. Validate Oracle database.</p> <p>Refer to the appendix: B.</p> <p>11. Startup standby database as a production database. Utilize the following command:</p> <pre>SVRMGR&gt; alter database open</pre> <p>It is also recommended to issue <i>alter system archive log all</i> command after opening; so, you can start shipping archives to the new standby database.</p> <p>12. Execute any post-open commands found in the create controlfile script such as renaming MISSING files and online tablespaces.</p> <p>(e.g. GracefulOpen.sql - Appendix C, Post Open Script Section)</p> <p>13. Clients reconnect to the standby site.</p> <p>14. Create standby controlfile by using the following commands:</p> <pre>SVRMGR&gt; alter database create standby controlfile as 'standby.ctl [reuse];</pre> <p>The new standby database controlfile must be placed in standard directory and shipped to the new standby database.</p> <p>15. Copy 'standby.ctl' to Primary site.</p> <p>16. Standby site becomes <b>Production</b> and is active.</p>

## Graceful Switchover and Switchback

<b>Primary Site</b> <b>Standby/Recovering Database</b>	<b>Standby Site</b> <b>Production/Active Database</b>
<p>17. Startup mount database as the new standby database. Issue the following commands:</p> <pre>SVRMGR&gt; startup pfile=init_standby.ora nomount SVRMGR&gt; alter database mount standby database</pre> <p>Again, these steps imply that there is a separate init.ora file for the standby role. This init.ora file must point to the new standby controlfile.</p> <p>Please refer to the Appendix: A.</p> <p>18. Rename data files and log files if required.</p> <pre>SVRMGR&gt; alter database rename file 'oldname' to 'newname';</pre> <p>All steps must be scripted and easily distinguishable by role: standby or production roles.</p> <p>19. Offline data files if required.</p> <pre>Alter database datafile 'filename' offline;</pre> <p>All steps must be scripted and easily distinguishable by role: standby or production roles.</p> <p>20. Validate Standby database.</p> <p>Please refer to the appendix B.</p> <p>21. Initiate recovery.</p> <pre>SVRMGR&gt; recover standby database or recover managed standby database</pre> <p>Or</p> <pre>SVRMGR&gt; recover from archive location standby database [until [time/cancel/change] ]</pre> <p>22. Apply archive logs</p>	
<b>Current Status: Standby/Recovering</b>	<b>Current Status: Production/Active Database</b>

database	
----------	--

### Graceful Switchback Steps

After a graceful switchover, the customer may eventually like to switch back to the primary site after its scheduled maintenance. The graceful switchback steps allow for switching of production and standby database roles without prompting an open resetlogs operation. Another scenario where a graceful switchback operation is helpful is after a disaster recovery. Eventually the primary site is fixed and a new standby database can then be created on the primary site. Once you fulfill the prerequisites described earlier, you can execute a graceful switchback. Why would you do this? Unfortunately, standby sites are usually subsets of the primary site in either hardware capability (reduced number CPUs or disks) or information. Standby sites tend to service mission critical applications while less critical applications data will not be safeguarded by the standby database. If the primary site can be fixed and returned to full service, customers will prefer to do a graceful switchback to the primary site.

Figure 5 gives the 7 basic steps to complete a graceful switchback. In this diagram, the standby site initially contains the production database and the primary site contains the standby database. At the conclusion the database roles change again.

Figure 5: Steps to Execute Graceful Switchback

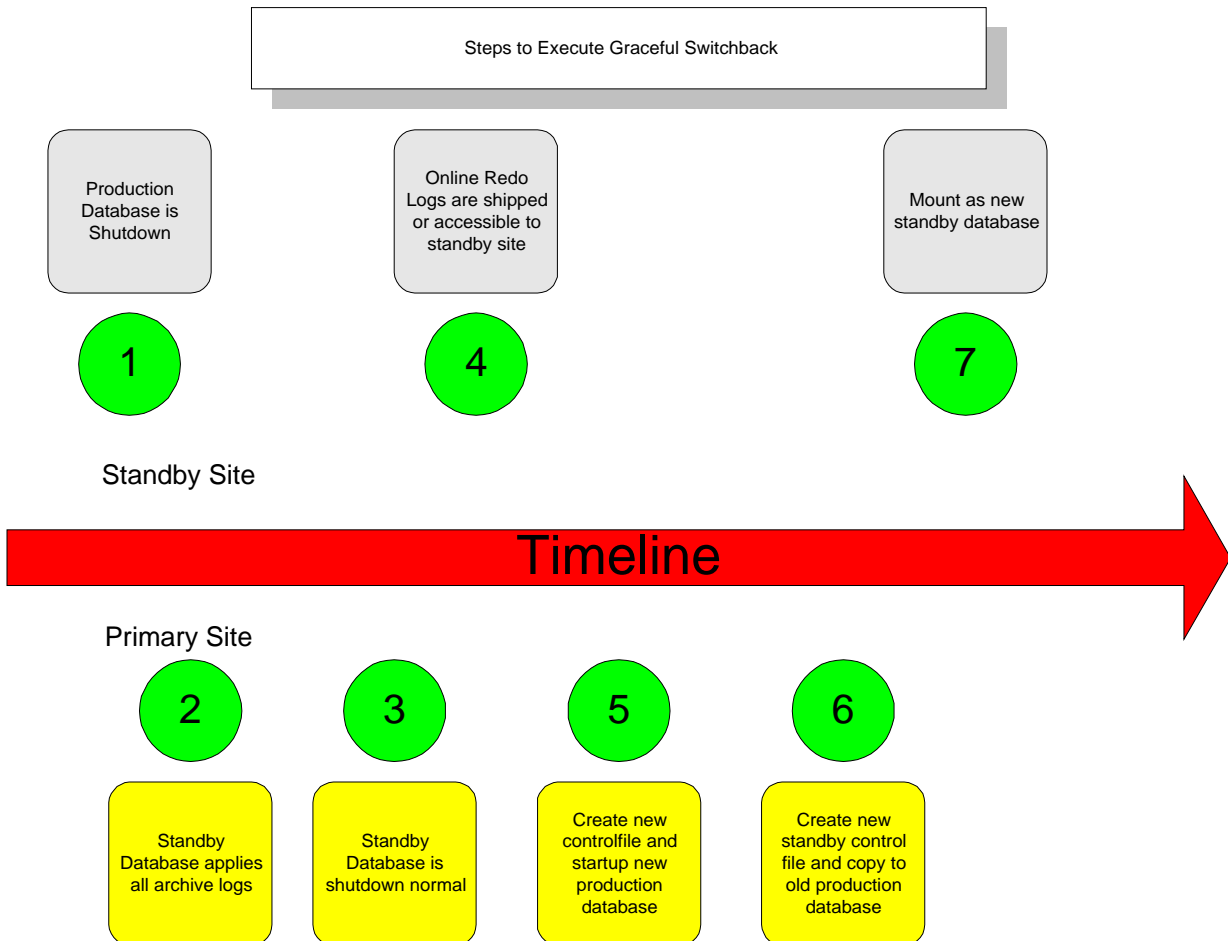


Figure 6, *Graceful Switchback*, is identical to figure 4, *Graceful Switchover*, except the source and target roles have changed. Table 3, *Graceful Switchback*, indicates that the steps are identical.

Figure 6 Graceful Switchback

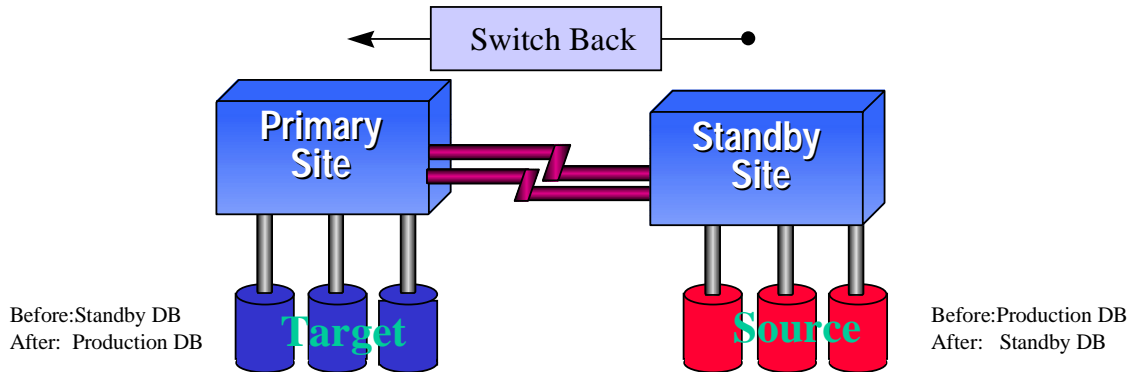


Table 3: Graceful Switchback Steps

Primary Site	Standby Site
Target Database	Source Database
Before: Standby Database in recovery mode.	Before: Production Database.
Follow steps 1-22 in the standby site column described in previous table.	Follow steps 1-22 described in the primary site column in previous table.
After: Production database	After: Standby Database

### Graceful Switchover and Switchback Explained

Graceful switchover and switchback steps are supported with Oracle versions 7.3 or higher; however extreme caution is required to validate and safeguard all steps. Since Oracle's current releases do not provide integrated scripts and commands to issue a graceful switchover and switchback, user errors that impacts the integrity of the primary or/and standby are common.

What makes the prescribed steps work and safe?

First we need to describe the supported mechanisms to initiate recovery and open a **standby** database. Next we can describe what we changed to execute a graceful switchover and/or switch back. Basically, we fooled the standby database into thinking it was a production database. Similarly, we fool the production database into thinking it was the standby database.

### ***Recovering and Opening a Standby Database***

A standby database has the following status:

- database is mounted,
- database is in recovery, and

recovery was initiated by either the following command:

- *recover standby database [until [time/cancel/change] ] or*
- *recover managed standby database*

Note: Only Standby databases in 7.3 or higher are supported because previous releases do not maintain and store data file status and controlfile changes in the online redo logs and archive redo logs. Oracle 7.3 and higher releases contain mechanisms to maintain standby databases and contain further checks and balances to ensure the integrity of the database. To use the procedures in this document, you need to upgrade to 7.3 or higher. Previous releases require strict control and monitoring of the production and standby databases for key events such as tablespace status changes or adding data files. For more information, please refer to *Standby Databases Prior to 7.3* white paper.

The above recover command explicitly tells Oracle that this is a standby database. Opening standby databases is done through an *alter database activate standby database* statement which implicitly does a resetlogs operation. The resetlogs operation prevents a graceful switchover and switchback. The next section describes how we can avoid the resetlogs operation.

### ***Converting a Standby Database into a Production Database***

Remember the goal is to switch the standby and production database's roles without jeopardizing the integrity of the database. In order for the standby database to assume the role of production, it has to **believe** and **contain** all the information required to start up the database as the normal production database. The database needs to be **restarted** with the following elements:

- current or production online redo logs,
- newly recreated controlfile, and
- existing data files.

The current online redo logs can be copied over to the standby database. This can only occur after the production database was shutdown normal successfully. Corruption can occur if you copy the online redo logs while the production database is still up or if the logs are still being written.

A new controlfile needs to be created using a modified create controlfile script. Previously, we recommend copying the current controlfile from the production database to standby database; however, we discovered situations in which the controlfile information was not consistent with that of the online data files. There are rare cases where these inconsistencies will prevent a successful switchover. To avoid these problems, Oracle now recommends recreating the controlfile with the *noresetlogs* option prior to opening the standby as the production database. The create controlfile script must be created from the production database and then modified for the standby database.

After restarting the standby database with the above prerequisites, the standby database will be mounted as the production database. Step 9, *recover database* command, will apply all the online redo log files and possibly archive logs in an OPS environment<sup>5</sup>. Finally, this database can be opened with the *alter database open* command. We have successfully avoided the *open resetlogs* operation.

---

<sup>5</sup> Please refer to the section on "Considerations for OPS".

### ***Converting Production Database into a Standby Database***

Converting the production database into a standby database is easier. You need to remount the production database with the following:

- standby controlfile and
- existing data files.

By creating a standby controlfile from the new production database, it can then be substituted with the current controlfiles. The data files are already present. The database now needs to be restarted by:

- *startup pfile=init\_standby.ora nomount;*
- *alter database mount standby database;*
- *recover standby database*

Standby database recovery will then be restarted. Now the production database has been converted into a standby database.

### **Best Practices**

As most people who implement standby databases know, the project can be huge. Enhancing your environment to include graceful switchover and switchback should not be detrimental or complex. However, the success of the standby environment still depends on how thoroughly the customer monitors and scripts these processes. The following are recommended best practices.

- Follow all steps in the graceful switchover and switchback tables. Validate and familiarized everyone with the procedures.
- Create a controlled environment that clearly separates standby and production database entities.
  - Standby and production database's init.ora files must exist on both sites.
  - Standby and production controlfiles must never be overwritten or mistaken. They must be separated, duplicated and clearly marked.
  - Standby and production online redo logs must be clearly marked. Never recommend overwriting existing logs; instead use different log names for each role.
- Automate and script all activity.
  - Maintenance, creation, refreshing, and activating standby databases must be scripted and duplicated on both primary and standby sites.
  - Other activity such as shipping of archives, renaming data and online redo log files must be scripted and duplicated on both sites. In some cases such as renaming files, separate scripts are required for production and standby databases.
  - Create a new controlfile script for the standby database every time there is a database change such as adding a data file or data file status change
- Rehearse and test scenarios thoroughly.
- Monitor system, database, and network logs to pro-actively catch problems or anomalies such as lack of disk space for archives or a database was successfully shut down.
- Periodic validation of the environment

## Graceful Switchover and Switchback

- Customized scripts to validate existence of controlfiles, online redo logs, data files.
- Customized scripts to validate the correct status of files. Data files are required to be online if they require recovery.
- Checks and database queries are provided in Appendix B to help maintain and validate the integrity of the production and standby databases.
- Validate that standby and production init.ora files and the parameters they contain are correct. Please refer to the Appendix A.
- Validate and check your create controlfile script. Ensure that you are using the noresetlogs option and that the script was created from the production database and modified for the standby database. Please refer to Appendix C.
- These procedures must not affect or harm normal standby database procedures.

## Appendix A - Init.ora Files

### *standby init.ora file*

One of the most common errors is bringing up the production or standby databases with the wrong init.ora file. Thereby, our recommendation is to have several init.ora files for each role your database plays. The init.ora file is essential since it dictates which controlfile will be used, how many resources will be allocated and eventually how fast your database will run. This section describes the essential init.ora parameters required in your standby init.ora file however; it is not intended to contain a comprehensive list of all parameters.

Standby database relevant only parameters:

db\_file\_standby\_name\_convert= "oldpath", "newpath" (7.3 only)

log\_file\_standby\_name\_convert="oldpath", "newpath" (7.3 only)

db\_file\_name\_convert= "oldpath", "newpath" (8.0 and 8.1)

log\_file\_name\_convert="oldpath", "newpath" (8.0 and 8.1)

Parameters that may need to be change:

control\_files= **pointing to standby controlfile or backup controlfile.**

log\_archive\_dest= new destination

background\_dump\_dest= new destination

core\_dump\_dest=new destination

user\_dump\_dest=new destination

ifile = point to new config.ora file

lock\_name\_space= if standby database is on the same machine as the primary (not advised)

Tuning standby database:

Since the standby database is essentially focused on database recovery, some customers optimize recovery by adjusting these common parameters.

db\_block\_buffers= increased if memory is available.

recovery\_parallelism= increased if memory is available and after testing.

log\_buffer= increased if memory is available.

db\_writers= increased if not using async io.

Gc\* parameters = set appropriately according to machine resources and application design

Parameters that need to be the same:

DB\_NAME

COMPATIBLE

DB\_FILES

***production Init.ora file***

It is crucial that production and standby init.ora files are separate and not mistaken for each other.

Parameters that are now irrelevant:

db\_file\_standby\_name\_convert= "oldpath", "newpath" (7.3 only)  
log\_file\_standby\_name\_convert="oldpath", "newpath" (7.3 only)

Parameters that need to be changed:

control\_files= **pointing to the production controlfiles**  
log\_archive\_dest= new destination  
background\_dump\_dest= new destination  
core\_dump\_dest= new destination  
user\_dump\_dest= new destination  
ifile= point to new config.ora file  
log\_archive\_dest (7.3 and 8.0 parameter) points to correct archive destinations  
log\_archive\_dest\_1 (8.1 parameter) points to correct archive destination  
log\_archive\_dest\_2 optionally point to standby archive destination or 2<sup>nd</sup> destination  
log\_archive\_dest\_state\_n enable

Parameters that need to be the same as standby database:

COMPATIBLE  
DB\_NAME  
DB\_FILES

## Appendix B - Queries and Checks

The following table describes the most basic and important checks.

Production Database	Standby Database
<p>Prerequisites:</p> <ol style="list-style-type: none"> <li>1. Monitor for nologged or unrecoverable events. - Refresh corresponding standby data files.</li> <li>2. Monitor for database error or corruption issues. - Stop standby recovery until fixed.</li> <li>3. Monitor for resetlogs or any missing or damaged archive logs or online redo logs - Recreate standby database</li> <li>4. Monitor for data file additions, tablespace or file status changes.                             <ul style="list-style-type: none"> <li>• Issue a new alter database backup controlfile to trace noresetlogs.</li> <li>• Modify trace file to be used for standby in case of graceful switchover.</li> <li>• Ship to standby site.</li> </ul> </li> </ol>	<p>Prerequisites:</p> <ol style="list-style-type: none"> <li>1. Refresh standby controlfile if production controlfile is recreated.</li> <li>2. Add any additional data files.</li> <li>3. Ensure that all archives are being applied promptly and without errors.</li> <li>4. Monitor recovery for any errors.</li> <li>5. Verify that the create controlfile script for graceful switchover has                             <ul style="list-style-type: none"> <li>• Noresetlogs option specified,</li> <li>• Correct datafile path names,</li> <li>• Correct logfile path names,</li> <li>• Log file names from all enabled threads,</li> <li>• Offline tablespaces that are offline normal</li> <li>• Rename missing files to appropriate pathnames</li> </ul> </li> </ol> <p>Please refer to Appendix C for create controlfile details.</p>

### *Queries to Validate Production Database*

#### **Queries to validate production database before opening the database.**

The following queries will help validate the integrity of your production or standby databases. Once you determine a problem, you must either start over and recopy the necessary files or call support for further assistance.

If you are on 8.1 or higher, you can attempt to open the database as read only and validate the database. To open the database as read only, execute the following command:

*alter database open read only;*

1. *select member, status from v\$logfile;*
  - all log files groups and members must be present
  - Status

## Graceful Switchover and Switchback

- must be blank
- Invalid - file is inaccessible. Need to check the path or file permissions of this file. The most common error is the file was not renamed.
- Stale - file is incomplete. This is acceptable if there is another member with blank status of the same group.

1. *select sequence#, status from v\$log;*

- the highest sequence number must have a status of current with the exception in an OPS environment.
- all other log sequences must have a status of inactive.
- status of unused implies that this log was just added.
- status of ACTIVE implies that log was copied while the database was still using it. You must copy these online redo logs again while the production database is shutdown successfully.

1. *select \* from v\$recover\_file;*

- If executed after the *recover database* command, this query must return zero rows.
- The most common errors are files requiring recovery, non-existent or offline. Attempt to correct this by comparing the file# with the corresponding name in v\$datafile and validating file existence, permissions, and file status.

1. *select \* from v\$datafile;*

- Validate that the statuses of each file is correct (online, system, offline). Files must not have the RECOVER status. If they do have the RECOVER status, **check if the files are fuzzy and recovery in v\$datafile\_header**.
- Validate the existence of all data files.
- Validate that all checkpoint\_change# are the same unless the files are read only or offline.
- Validate that the unrecoverable\_change# < change# in v\$backup for the same file. If the unrecoverable\_change# is greater, the data will be corrupted or unusable in this database. This may have been intentionally and expected if unrecoverable activity has occurred in the production database without resynching the standby database.

5. *select \* from v\$datafile\_header (O8 only)*

- Validate correct status for each data file.
- ERROR column must be blank if there are no problems.
- RECOVER=no implying no recovery is needed.
  - If yes, **more recovery is required**.
- FUZZY = no
  - if FUZZY=yes, **more recovery is probably required**.
  - the data file requires more recovery because the online redo logs and controlfiles were not compatible with the data files. Files can have three types of fuzziness: media recovery, hot backup and online fuzzy. In all cases, the file is inconsistent until the fuzzy bit is cleared.

### *Finding Database Resetlogs Version*

In order for graceful switchover and switchback to work, both databases must have the same resetlogs version. Unfortunately, versions prior to Oracle8 does not provide an easy method to access this information. We have listed two ways to access this information.

1. Oracle8 ONLY: *select resetlogs\_change#, resetlogs\_time, open\_resetlogs from v\$database;*
  - resetlogs\_change# is the change number at open resetlogs. It is zero if the database never undergone a resetlogs operation.
  - resetlogs\_time is the timestamp of the open resetlogs.
  - open\_resetlogs indicates whether the next database open requires a resetlogs operation.
  
1. You can also access this information by dumping the controlfile searching for the appropriate text.
  - dumping controlfile: *alter session set events 'immediate trace name controlf level 10';*
  - a trace file will be created in your *user\_dump\_dest*.
  - For each controlfile trace file, search for “Resetlogs scn”. This is the database resetlogs version. The target and standby databases’ resetlogs scn must match.
  - example: *grep “Resetlogs scn” control\_file.trc*

output: Incmplt recovery scn: 0x0000.00000000 **Resetlogs scn: 0x0000.000039d1** count:0x12edd924  
This database has executed a resetlogs operation.

## Appendix C - Create Controlfile Script

Validate existence and accuracy of the *create controlfile* script. You can create the create controlfile script by issuing *alter database backup controlfile noresetlogs* command from the production database. The trace file will be created in your user dump destination. The output of the trace file needs to be edited and shipped to the standby database. He's a sample of the trace files and the portions that were edited.

Do not create the script from the standby database since offline and read only tablespaces are not handled properly if generated from a backup or standby database. Please refer to bug 1034871.

Sample Create Controlfile Script generated by *alter database backup controlfile to trace noresetlogs* command from the production database. Renamed the trace file to *CreateProductionCtl.sql*

Sample *CreateProductionCtl.sql* with directions

```
# The following commands will create a new control file and use it to open the # database.
# Data used by the recovery manager will be lost. Additional logs may
# be required for media recovery of offline data files. Use this
# only if the current version of all online logs are available.
```

*# Comment out all trace file header information*

```
STARTUP NOMOUNT
CREATE CONTROLFILE REUSE DATABASE "CRASH" NORESETLOGS ARCHIVELOG
```

*# Validate all settings*

```
MAXLOGFILES 10
MAXLOGMEMBERS 2
MAXDATAFILES 30
MAXINSTANCES 2
MAXLOGHISTORY 7196
```

```
LOGFILE
```

*# Validate and correct log file names and file paths. All  
# log files from every enabled thread (active OPS instance) should  
# be included.*

```
GROUP 1 '/u02/oracle/lto/standby/redoCrash1.dbf' SIZE 50K,
GROUP 2 '/u02/oracle/lto/standby/redoCrash2.dbf' SIZE 50K,
GROUP 3 '/u02/oracle/lto/standby/redoCrash3.dbf' SIZE 50K,
GROUP 4 '/u02/oracle/lto/standby/redoCrash5.dbf' SIZE 50K,
GROUP 5 '/u02/oracle/lto/standby/redoCrash7.dbf' SIZE 50K
```

```
DATAFILE
```

*# Validate and correct all data file names and file paths. Only data files  
# from online tablespaces are included. Read only tablespaces are also  
# not included.*

```
'/u02/oracle/lto/standby/sysCrash.dbf',
'/u02/oracle/lto/standby/rbsCrash01.dbf',
'/u02/oracle/lto/standby/usersCrash03.dbf'
```

```
CHARACTER SET US7ASCII
```

*# Take files offline to match current control file.*

```
ALTER DATABASE DATAFILE '/u02/oracle/lto/standby/usersCrash03.dbf' OFFLINE;
```

*# Recovery is required if any of the datafiles are restored backups,  
# or if the last shutdown was not normal or immediate.*

## Graceful Switchover and Switchback

```
# Comment out recover and add to GracefulOpen.sql script
# Comment out all commands from this point on and add to GracefulOpen.sql
# RECOVER DATABASE

# All logs need archiving and a log switch is needed.
# ALTER SYSTEM ARCHIVE LOG ALL;
# Database can now be opened normally.
# ALTER DATABASE OPEN;

# Files in normal offline tablespaces are now named.
# Validate and correct all data file names and file paths.
# ALTER DATABASE RENAME FILE 'MISSING00003'
# TO '/u02/oracle/lto/standby/usersCrash01a.dbf';
# ALTER DATABASE RENAME FILE 'MISSING00006'
# TO '/u02/oracle/lto/standby/usersCrash01b.dbf';

# Files in read-only tablespaces are now named.
# ALTER DATABASE RENAME FILE 'MISSING00004'
# TO '/u02/oracle/lto/standby/rbsCrash02.dbf';

# Online the files in read-only tablespaces.
# ALTER TABLESPACE "USERS2" ONLINE;

# No tempfile entries found to add.
# Validate and correct all data file names and file paths.
```

## Appendix C - Post Open Script

Sample GracefulOpen.sql with directions  
This script can be executed after opening the database.

```
# Comment out the next 2 commands if already executed in Step 9.
# RECOVER DATABASE
# All logs need archiving and a log switch is needed.
# ALTER SYSTEM ARCHIVE LOG ALL;

# Comment out the alter database open if already executed in Step 11.
# Database can now be opened normally.
# ALTER DATABASE OPEN;

# Files in normal offline tablespaces are now named.
# Validate and correct all data file names and file paths.

ALTER DATABASE RENAME FILE 'MISSING00003'
TO '/u02/oracle/lto/standby/usersCrash01a.dbf';
ALTER DATABASE RENAME FILE 'MISSING00006'
TO '/u02/oracle/lto/standby/usersCrash01b.dbf';
```

## Graceful Switchover and Switchback

```
# Files in read-only tablespaces are now named.  
# Validate and correct all data file names and file paths.  
ALTER DATABASE RENAME FILE 'MISSING00004'  
TO '/u02/oracle/lto/standby/rbsCrash02.dbf';  
  
# Online the files in read-only tablespaces.  
ALTER TABLESPACE "USERS2" ONLINE;  
# No tempfile entries found to add.  
# Validate and correct all data file names and file paths.  
# There may be additional offline immediate tablespaces that will require  
# recovery.
```

## Appendix D - Considerations for OPS

When the primary database is Oracle Parallel Server (OPS), graceful switchover and switchback procedures are slightly different. Standby database recovery remains essentially the same; however, Oracle will detect any new enabled or disabled threads during recovery. Recovery will then merge all enabled threads and apply them to the standby database as a single stream. While applying one log group, Oracle may wait for the availability of other log groups from different threads. Recovery will always look at specific SCN ranges of all enabled threads.

Thread 1	Archive Log 65 SCN range 100-255	Archive Log 66 SCN range 255 -700	Online Log 67 SCN range 700-919
Thread 2	Archive Log 14 SCN range 1-101	Archive Log 15 SCN range 101-199	Online Log 16 SCN range 199-1000

Using the above 2 threads as an example, standby database recovery may work like the following:

- Merge thread 2, archive 14 with thread 1, archive 65 - complete applying archive 14, thread 2.
- Merge thread 2, archive 15 with thread 1, archive 65 - complete applying archive 15, thread 2.
- Wait on online redo log 16 to be archived - did not completely apply archive 65, thread 1.

In a graceful switchover or switchback scenario, the new production database may need to apply archive logs prior to opening the production database. For example, if the primary database was shutdown using the above example and switchover commenced, the new production database would request archive logs 65 and 66 during recovery.

For OPS, we recommend these additional best practices.

Before switchover and shutdown of the production database, follow these additional procedures from the production database

- Save the output of *select \* from v\$log\_history order by recid;*
- Save the output of *select a.member, b.group#, b.thread#, b.sequence#, b.status, b.first\_change#, b.first\_time from v\$log b, v\$logfile a where a.group#=b.group# order by thread#, sequence#;*
- Save the output of *select \* from v\$thread;*
- Issue the following command, *alter system archive log current* (Step 2 from Table 2)

While attempting to switch over,

Follow the same best practices prescribed in this paper.

- Additionally, ensure that all redo log groups from all enabled threads are in the create controlfile script.
- Recover database may require archive logs. Please refer to the output from *v\$log\_history*, *v\$log*, *v\$logfile*, *v\$thread* to determine which logs to apply. Because the standby controlfile has been created

## Graceful Switchover and Switchback

using `CREATE CONTROLFILE NORESETLOGS`, recovery cannot prompt for log sequence number and thread; it does not have this information. It prompts only for System Change Number (SCN).

## **Acknowledgments**

Bill Bridge, Oracle Corporation

Mark Smith, Oracle Corporation

Rick Anderson, Oracle Corporation

Alok Pareek, Oracle Corporation

Carol Colrain, Oracle Corporation

Anna Logan, Oracle Corporation

Bill Lee, Oracle Corporation

Wei Hu, Oracle Corporation

## Relevant Bugs

Bug 1034927 8.1 RDBMS 8.1 RECOVERY PRODIG-5 PORTID-453 ORA-1207

Abstract: DATAFILES FROM STANDBY CAN MAKE PRIMARY CONTROLFILE LOOK LIKE A BACKUP

If the primary controlfile is copied to a standby for a graceful switchover to the standby, ORA-1207 may be signalled when reading datafile headers. If a datafile backup is taken at the standby and copied to the primary for recovery, ORA-1207 may be signalled when reading the backup datafile header. This happens when the standby does more controlfile transactions than the primary. The controlfile transaction count is recorded in the datafile header on every write. If the counter is greater than the counter in the controlfile then the presumption is that the controlfile must be a backup. However if the last write to the datafile header was done at the standby this check may fail because the standby controlfile counter is higher than the primary's. One work around is to recreate a standby controlfile from the primary and use it for recovery at the standby just before copying the backups to the primary, or attempting graceful switch over. Another work around is to do a CREATE CONTROLFILE NORESETLOGS to make a new primary controlfile. The fix is to eliminate the check for ORA-1207. The checkpoint counters in the controlfile and datafile headers accomplish the same thing except that they always work right. Note that there are two checks for 1207 that must be eliminated.

Bug 1034871

Abstract: BACKUP CONTROLFILE TO TRACE OUTPUT IS WRONG IF USING BACKUP/STANDBY CONTROLFILE

If the currently mounted controlfile is either a backup or a standby controlfile then the script produced by ALTER DATABASE BACKUP CONTROLFILE TO TRACE will be wrong. In particular, datafiles from read only tablespaces will be listed in the DATAFILE clause rather than left out and renamed after open. There may be other problems since the command was not designed to work with a backup controlfile. This problem was noted when attempting to do a graceful switch to a standby -i.e. with no redo loss and no resetlogs. A CREATE CONTROLFILE NORESETLOGS command can be used at the standby to allow online logs copied from the primary to be applied to the standby, and to open the standby without RESETLOGS. However the BACKUP CONTROLFILE TO TRACE command to construct the script must be executed at the primary not the standby. If the read only datafiles are included in the DATAFILE clause everything will appear to work fine. The read only tablespaces will be available and can be made read write. However if they are left read only and their is a subsequent resetlogs (maybe years later) the read only tablespaces will no longer be accessible.