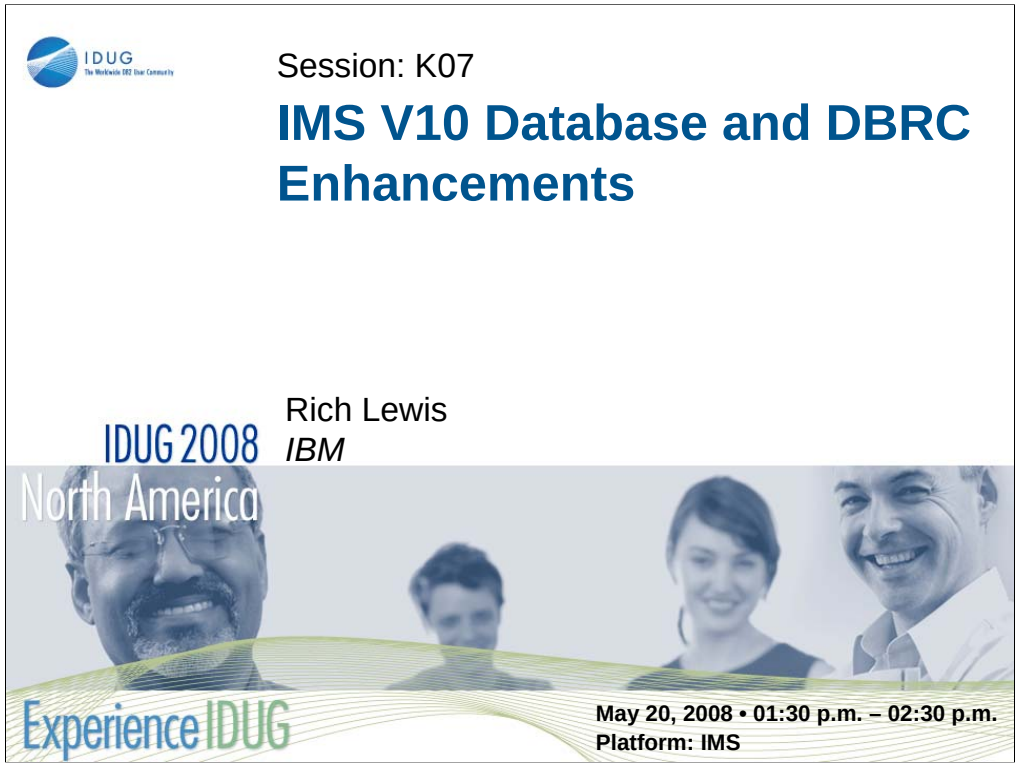
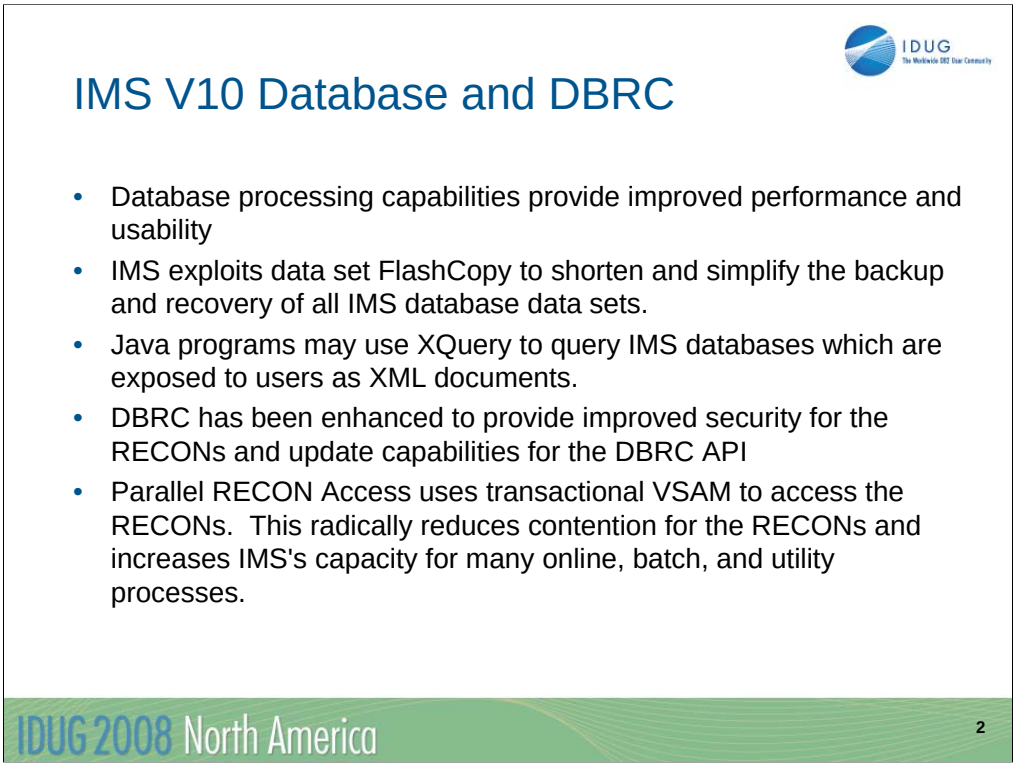


IMS Version 10 adds many enhancements to this premier database and transaction manager. In this session you will learn:

- How you can integrate FlashCopy into your image copy process
- How your Fast Path systems can take advantage of larger processors
- How you can use XQuery with IMS databases to find data within XML documents
- How both your experienced and junior DBAs can benefit from DBRC enhancements
- How security can authorize access to DBRC RECONs for readers versus updaters
- How you can eliminate RECON access bottlenecks with Parallel RECON Access
- And more!



Database enhancements:

•HALDB ILDS Rebuild Enhancement

•Image Copy 2 Fast Replication

•Fuzzy User Image Copy Support

•ACBGEN Exploitation of Storage Above 16M

•Fast Path Command Enhancements

•Fast Path Capacity Enhancements

DBRC enhancements:

•DBRC Timestamp Precision

•RECON READONLY Access

•Improved SAF Support for RECONs

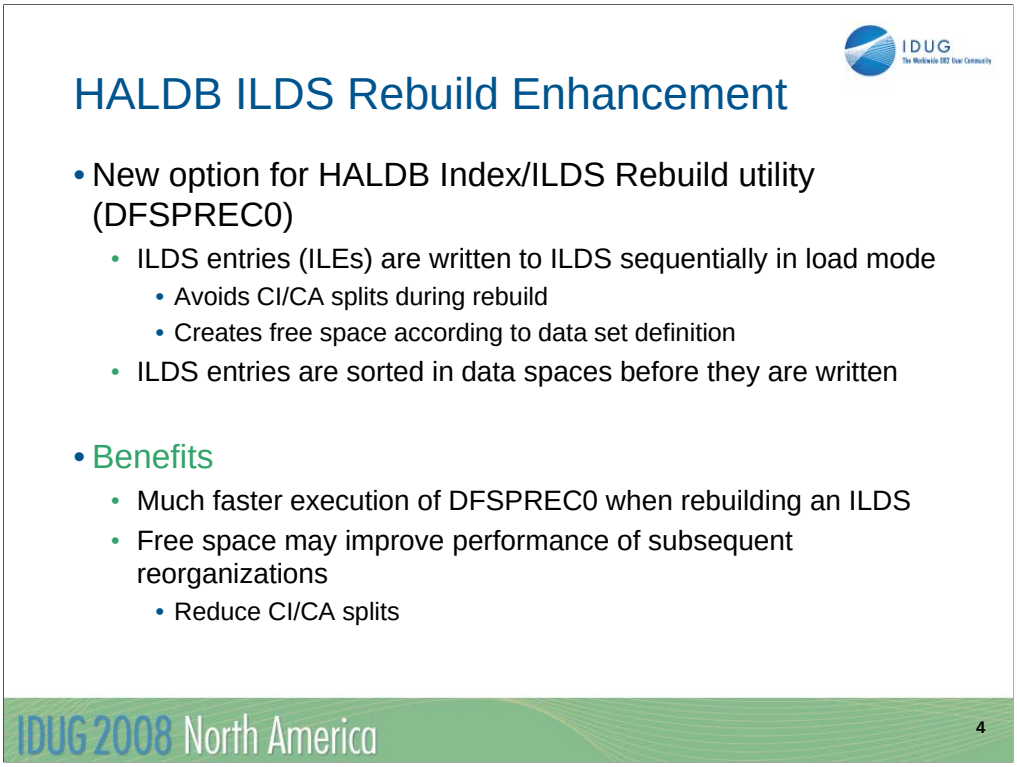
•DBRC API Enhancements

•Parallel RECON Access

Database Enhancements

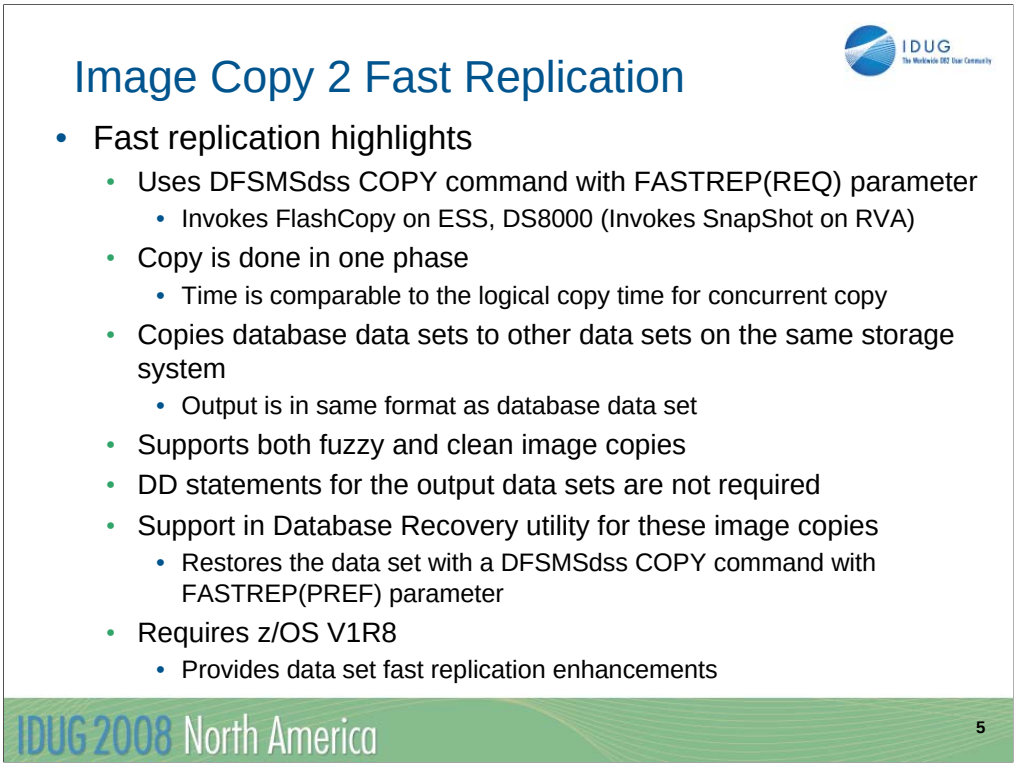
- HALDB ILDS Rebuild Enhancement
- Image Copy 2 Fast Replication
- Fuzzy User Image Copy Support
- ACBGEN Exploitation of Storage Above 16M
- Fast Path Command Enhancements
- Fast Path Capacity Enhancements

This is a list of the database topics that are explained on the following pages.



The HALDB Index/ILDS Rebuild utility (DFSPREC0) is used to rebuild either a PHIDAM primary index, an ILDS, or both for a partition. This enhancement applies only to the rebuilding of the ILDS. When an ILDS is rebuilt, the utility reads the partition database data sets looking for segments which are targets of secondary indexes or logical relationships. An indirect list entry (ILE) is written to the ILDS for each target that is found. In previous releases, these writes are done when the target segments are found. This means that the writes are random. They must be done in update mode. The enhancement provides a new way of writing the ILEs. When using the enhancement, ILE entries are not immediately written to the ILDS. Instead, they are stored in data spaces, sorted in ILE key sequence, and then written. The writes are done in load mode. These writes are sequential. Load mode avoids CI/CA splits.

The use of the new option could significantly reduce the elapsed time required for rebuilding ILDSs. The creation of free space in the ILDS could improve the performance of subsequent HD Reload or Online Reorganization executions since the free space should reduce the need for CI/CA splits.



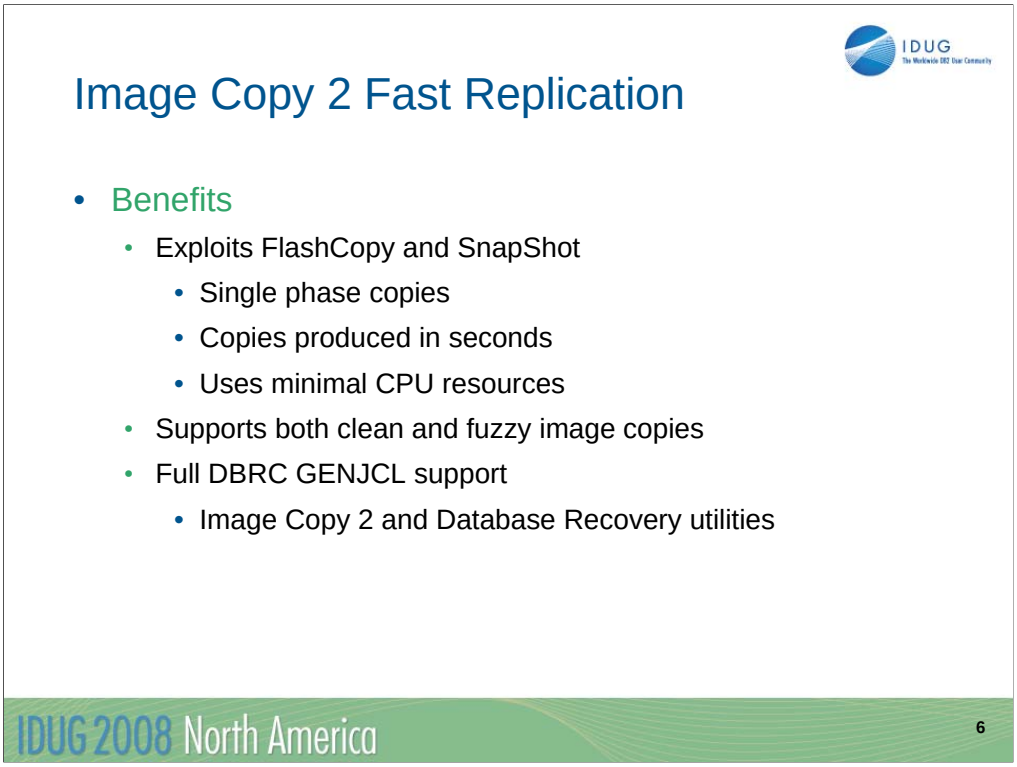
IMS V10 adds fast replication support. Fast replication is a DFSMSdss function that exploits the FlashCopy capability of the Enterprise Storage Server (ESS or Shark) or TotalStorage DS8000 or the SnapShot capability of the IMS RAMAC Virtual Array (RVA). Fast replication also can be used with OEM storage devices which support the DFSMSdss Fast Replication interface.

Unlike the concurrent copy process, the fast replication copy process is not split into a logical copy phase and a physical copy phase. If you are creating a clean image copy, the database is unavailable for the duration of the copy process; however, the fast replication process is comparable in speed to the logical copy phase of the concurrent copy process.

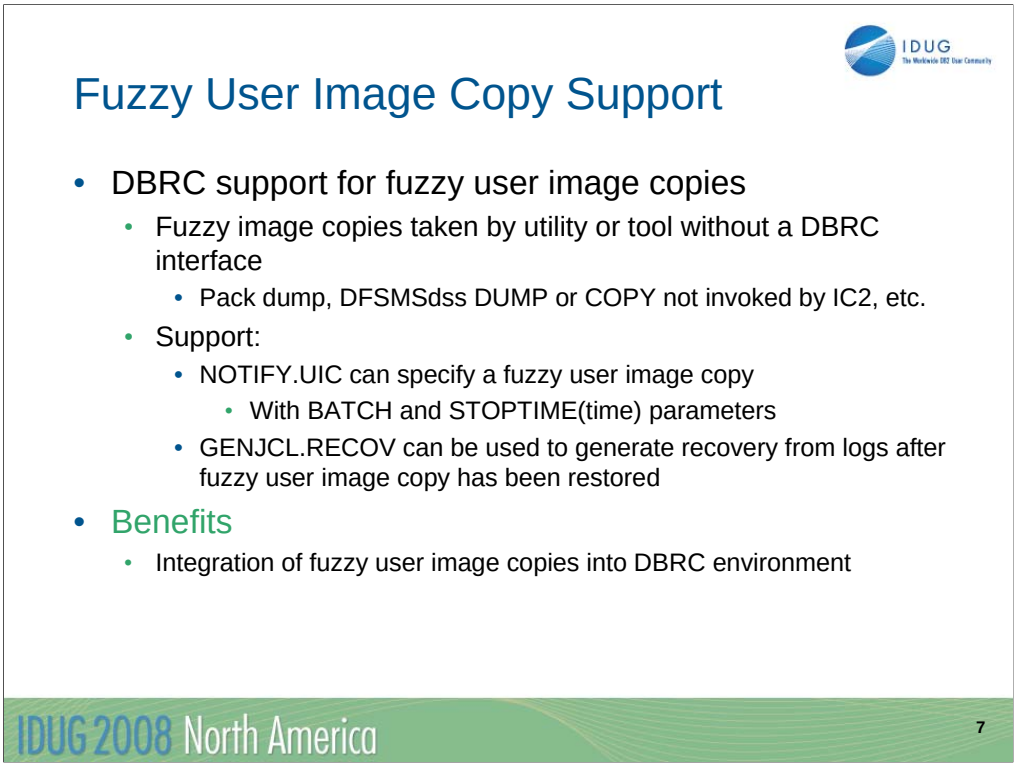
The input data set (the database data set) and the output data set (the image copy data set) must reside on the same storage system. The image copy is in the same format as the database data set.

Both fuzzy and clean image copies are supported.

This support is provided in the Image Copy 2 (DFSUDMT0) utility and the Database Recovery (DFSURDB0) utility. They use data set level Fast Replication functions of DFSMSdss which are only provided by z/OS V1R8.



Fast Replication exploits FlashCopy technology (or SnapShot for RVA devices). This produces copies which typically take only a few seconds while using minimal CPU resources since the copy is a storage subsystem function. Fast Replication can be used to produced either clean or fuzzy image copies. DBRC has been enhanced so that GENJCL commands may be used to create IC2 and Database Recovery JCL which invokes Fast Replication.



IMS V10 has added support for fuzzy user image copies. User image copies are copies of database data sets that are done without a DBRC interface. Typically, these are done with utilities that are not IMS utilities. In previous releases IMS supported clean user image copies. IMS V10 adds support for image copies taken while the database data set is being updated.

The support has two parts. First, the NOTIFY.UIC command has been extended to include a specification of a fuzzy image copy, sometimes called a concurrent image copy. Second, the GENJCL.RECOV command has been extended to generate the appropriate JCL to include the correct logs for recoveries using fuzzy user image copies.

Users now have the flexibility to use non-IMS utilities to produce fuzzy user image copies and can use GENJCL to create recovery jobs after restoring these image copies.

ACBGEN Use of Storage Above 16M

- Previous releases were limited in the number of PCBs per PSB
  - Limitation was due to use of "below the line" storage
    - PSBs with more than approximately 500 PCBs could result in S80A abend
- IMS V10 ACBGEN allocates most of it working storage above the 16M line
  - Eliminates these out-of-storage abends
- Benefit
  - Allows up to 2500 PCBs per PSB

ACBGEN has been modified to allocate most of its working storage above the 16M line. In previous releases, this working storage was allocated below the line. Very large PSBs could require more storage than was available below the line. Requests for more storage than was available resulted in S80A abends with RC=10 in previous releases. Since V10 allocates storage above the line, it is not limited to the available storage below the line. This should eliminate these abends. Very large PSBs, including those with up to 2500 PCBs, are handled by ACBGEN in V10.

The limitations of 2500 PCBs per PSB and 30,000 SENSEGs per PSB are enforced by the macros used in PSBGENs.

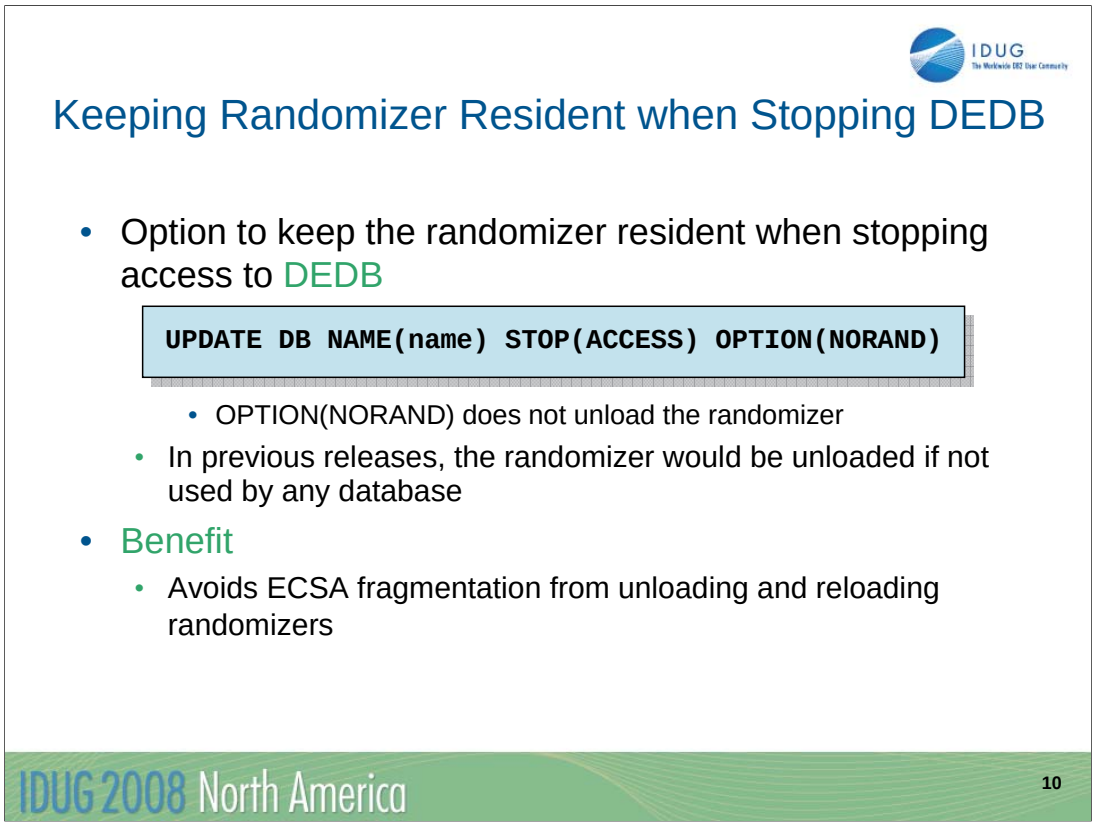
Starting All Areas with UPD DB Command

- Option to start all areas when starting a DEDB

```
UPDATE DB NAME(name) START(ACCESS) AREA(*)
```

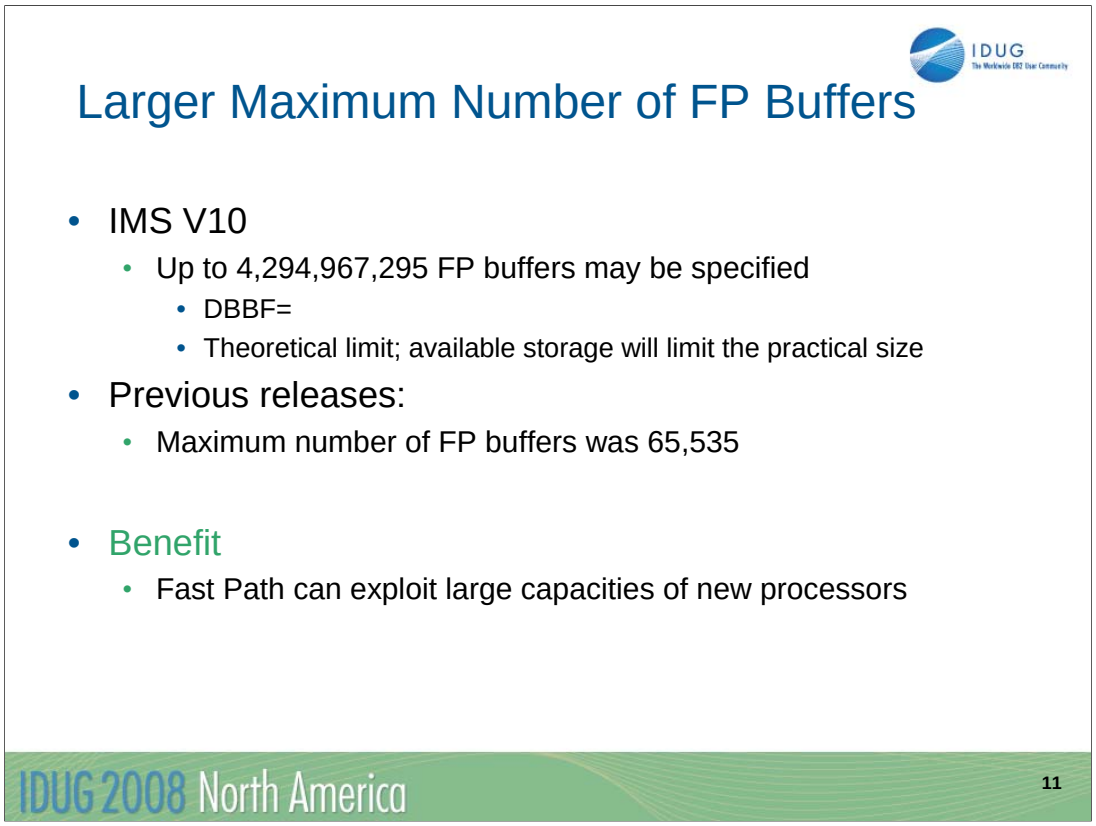
  - AREA(*) starts all areas of the database
  - In previous releases areas had to be started separately
- Benefit
  - Separate UPDATE AREA commands are not required for each area

The UPDATE DB START(ACCESS) command for DEDBs has a new option. This enhancements allows you to start all of the areas in a database when you also start the database. The only valid specification for the AREA parameter is AREA(*). In previous releases areas could only be started with a command that specified the area name. This simplifies starting of DEDB areas in V10.

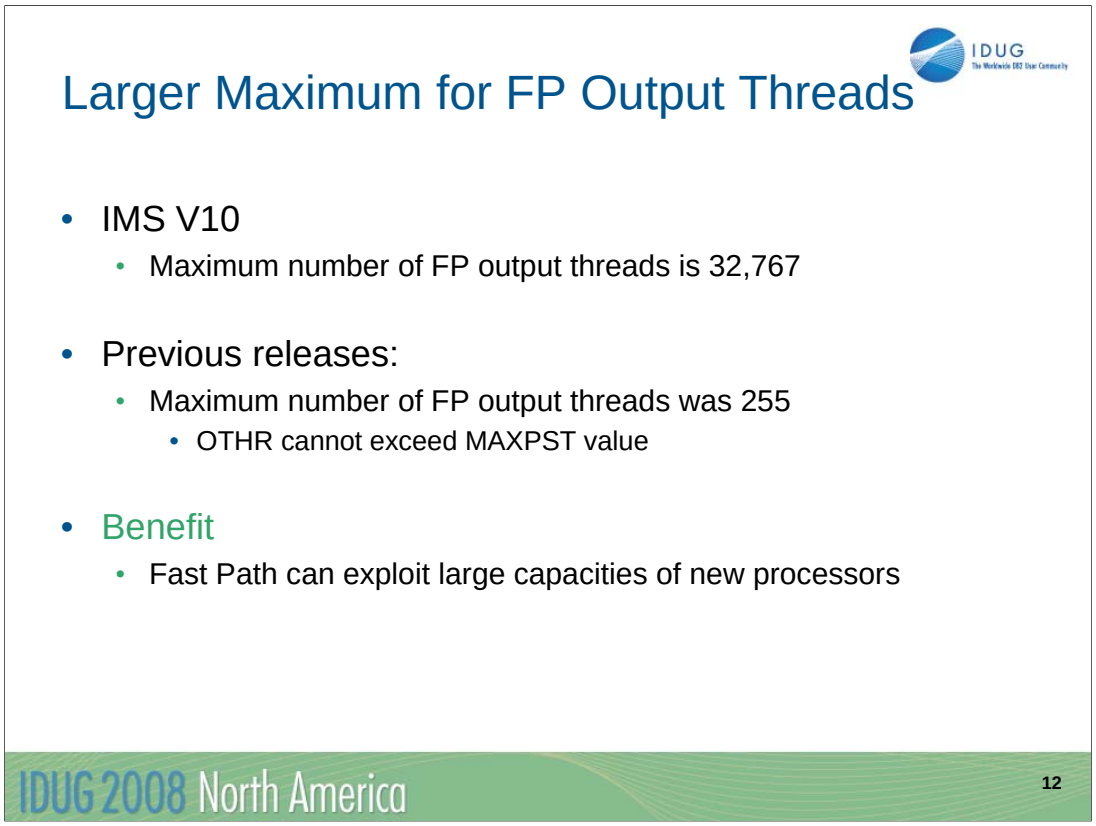


In previous releases if you made a DEDB inaccessible, its randomizer was unloaded from memory if the randomizer was not being used by any other DEDB. DEDBs are made inaccessible with either a /DBR DB command or an UPDATE DB STOP(ACCESS) command.

IMS V10 adds an OPTION(NORAND) parameter to the UPDATE DB STOP(ACCESS) command. This causes IMS to keep the randomizer module in memory. This helps to avoid fragmentation of ECSA which can occur when modules are unloaded and reloaded.



The maximum number of fast path buffers that can be used has been increased from 65,535 to a theoretical value of 4,294,967,295. This is a theoretical value since a system is unlikely to have the storage for so many buffers. Any value up to the theoretical limit may be specified on the DBBF= execution parameter for the online system. In previous releases you could specify the number of Fast Path buffers on the FPCTRL macro. This macro is not used in IMS V10. The DBBF= execution parameter must be used to specify these buffers in V10.



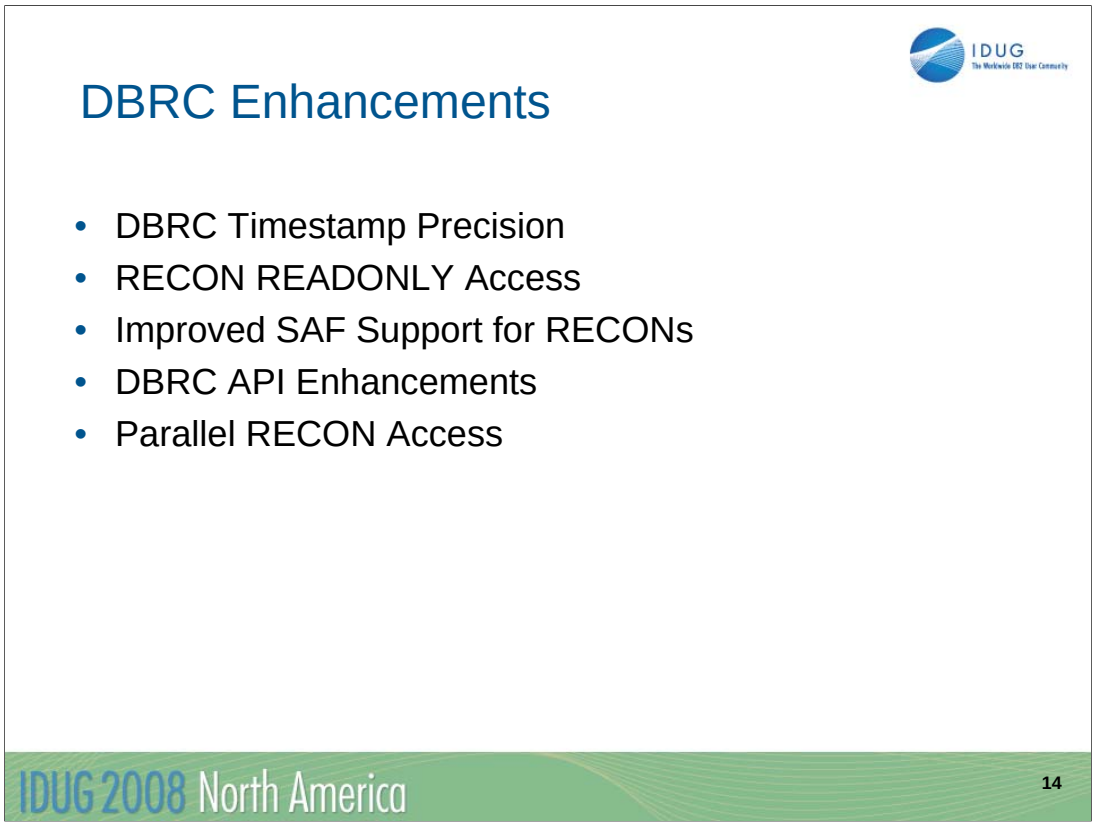
In previous releases the maximum number of Fast Path output threads was the smaller of the MAXPST value or 255. In V10 the maximum is the 32,767. Output threads are used to write Fast Path database updates asynchronously allowing IMS dependent regions or threads to continue with other work.

The increase in output threads could be desirable as processor capacities increase.

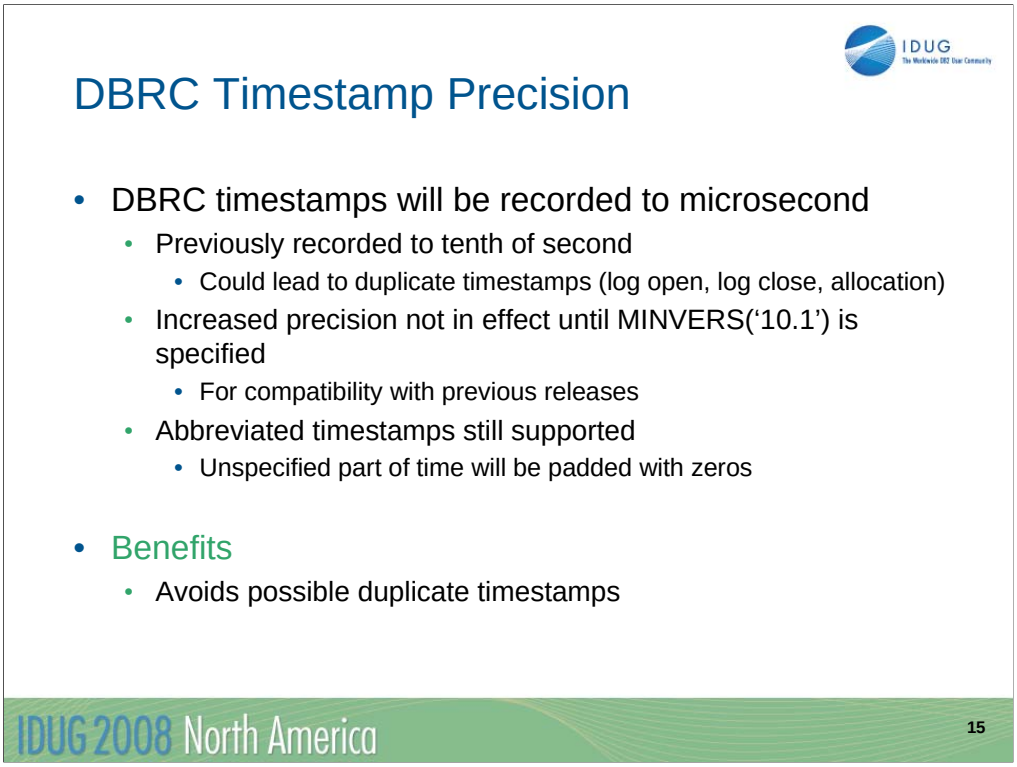
DBFCONT0 Converted to Multiple Modules

- Previous releases:
  - Most Fast Path control blocks and buffers were placed in one module (DBFCONT0)
    - DBFCONT0 contents included: ECNTs, MSDBs, MSDB blocks, Buffer Headers (DHMRs), Buffers, DEDB blocks, output threads, and BALGs
- IMS V10
  - These Fast Path control blocks and buffers are placed in five modules
- Benefit
  - More efficient use of ECSA storage
    - Required contiguous area is smaller

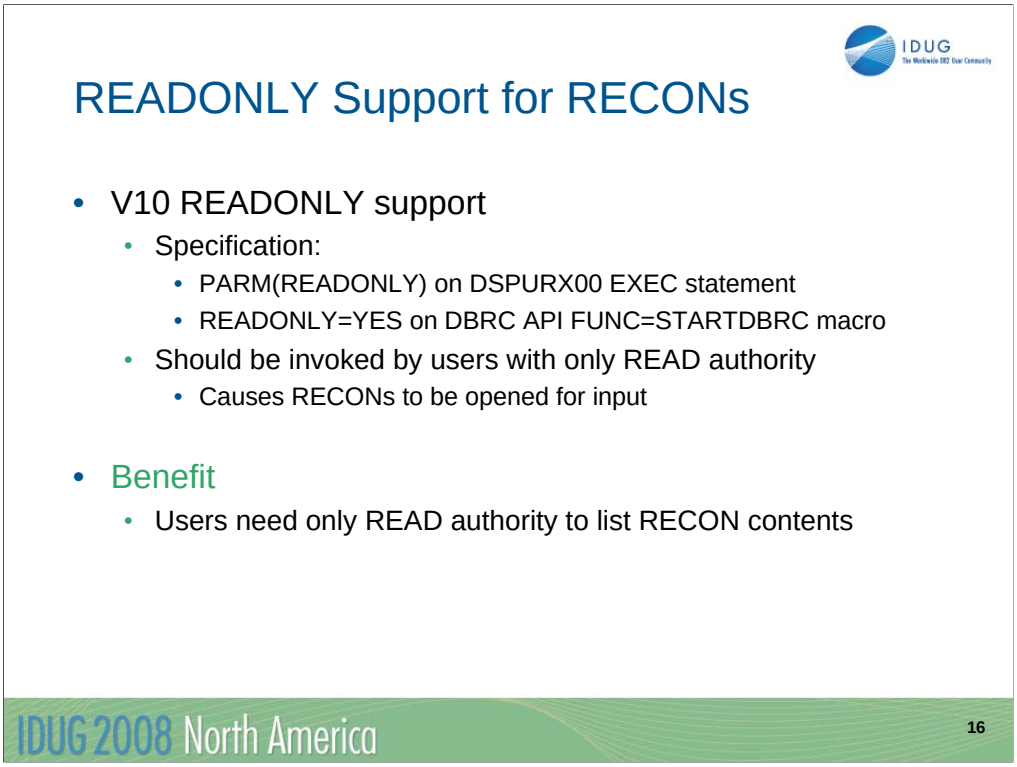
In previous releases most Fast Path control blocks and buffers were created in one module, DBFCONT0. This module resides in ECSA and it could be very large. IMS V10 places these control blocks and buffers in five modules. This allows installations to more efficiently use ECSA storage since smaller contiguous areas of ECSA may be used.



This is list of the DBRC topics that will be explained.

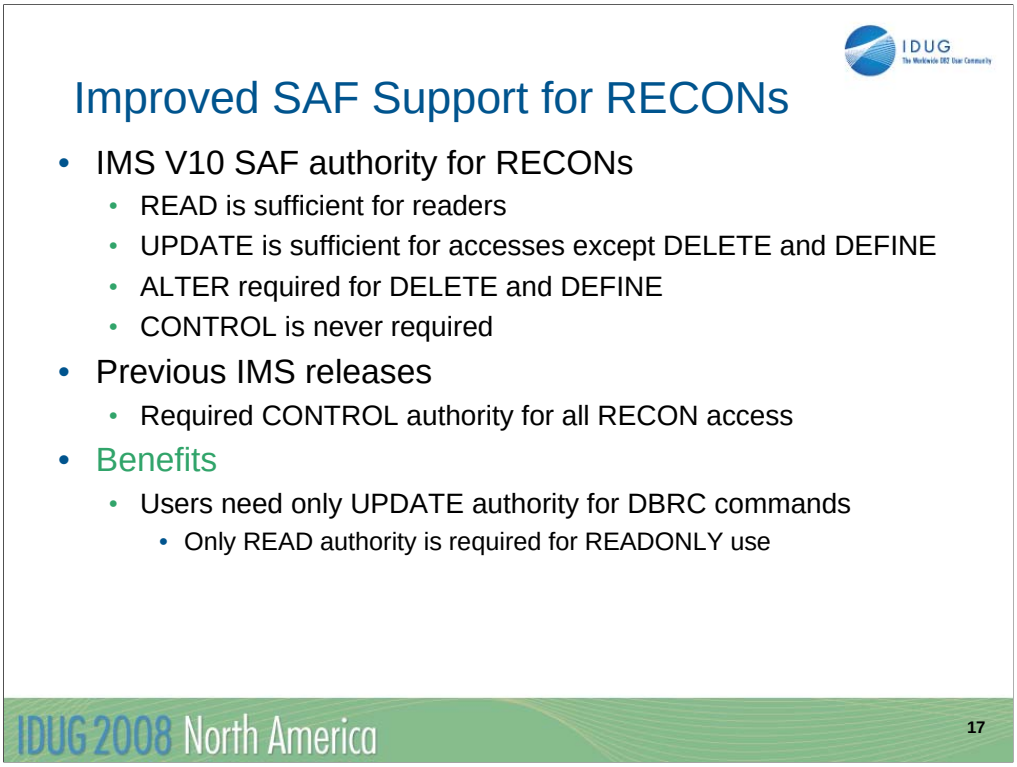


IMS V10 will record timestamps to the microsecond when MINVERS('10.1') is in effect. Previous releases record timestamps to the tenth of a second. When a user specifies a timestamp, it may be abbreviated. That means that times to the microsecond do not have to be specified. Unspecified parts of the time are padded with zeros. The finer granularity of timestamps can avoid some potential duplicate timestamps which cause problems in DBRC processing.

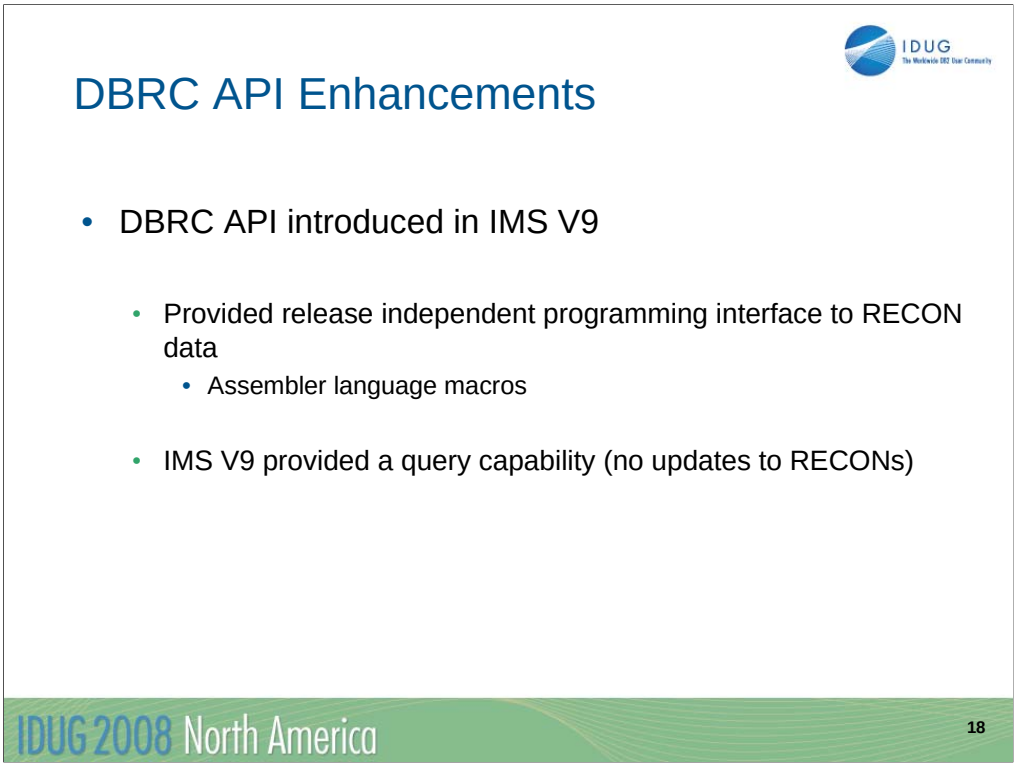


IMS V10 adds READONLY support for the RECONs. This is invoked with PARM(READONLY) on the EXEC statement for the DBRC utility (DSPURX00) or by specifying the new READONLY=YES parameter on the DBRC API FUNC=STARTDBRC macro. When READONLY is specified, the RECONs are opened for read.

This means that only READ authority is required in SAF (RACF). Users do not have to be given CONTROL authority to list RECON records.

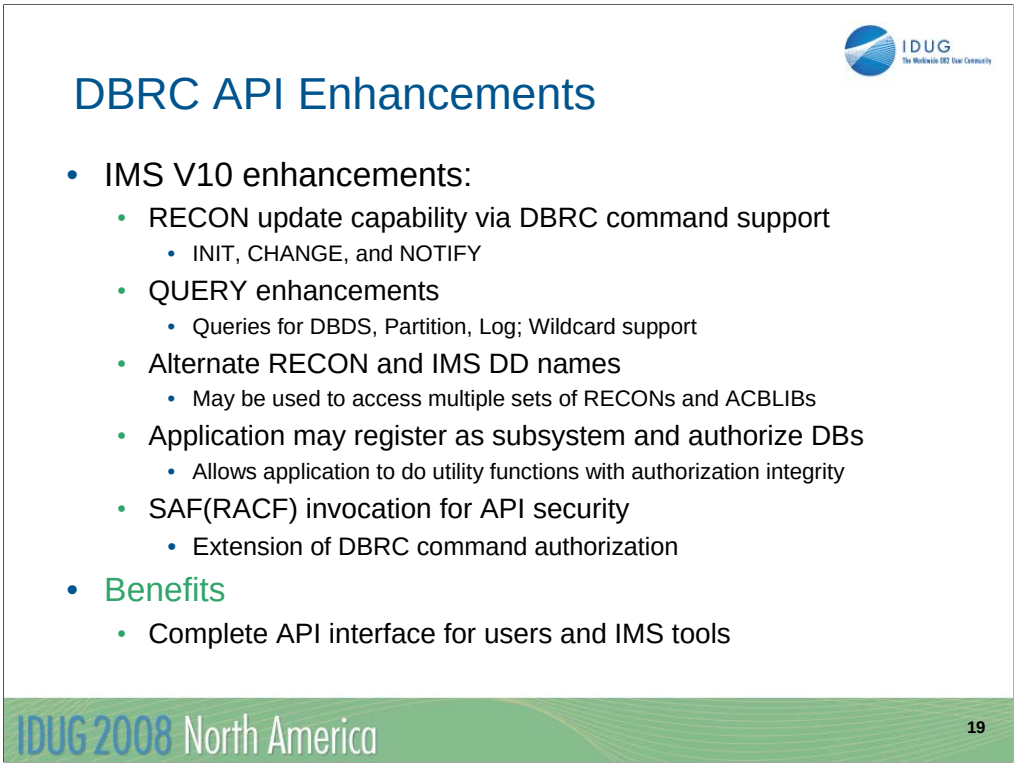


IMS V10 has made another change to open for RECON data sets. Due to this change, CONTROL does not have to be specified for users who update the RECONs. Only UPDATE authority is required. Of course, ALTER is still required for users who DELETE and DEFINE the data sets. In previous releases DBRC opened the RECONs with CONTROL specified in the VSAM ACB for the RECONs. This required CONTROL authority for open. In IMS V10 the open has changed. If READONLY is not specified, the open is done for update but CONTROL is not specified in the ACB. This means that only UPDATE authority is required.



The DBRC API was introduced in IMS V9. It provides a release independent assembler language interface for querying the data in the RECONs.

This interface may be used as an alternative to parsing the output of a LIST.RECON command or to reading the RECONs using VSAM macros. The output of the LIST.RECON typically changes with each release of IMS. The contents of RECON records often change with IMS releases. Both of these factors make it necessary to change programs with IMS release changes when these techniques are used. The DBRC API does not have these problems. Its output is consistent across IMS releases.



The DBRC API has been enhanced in IMS V10.

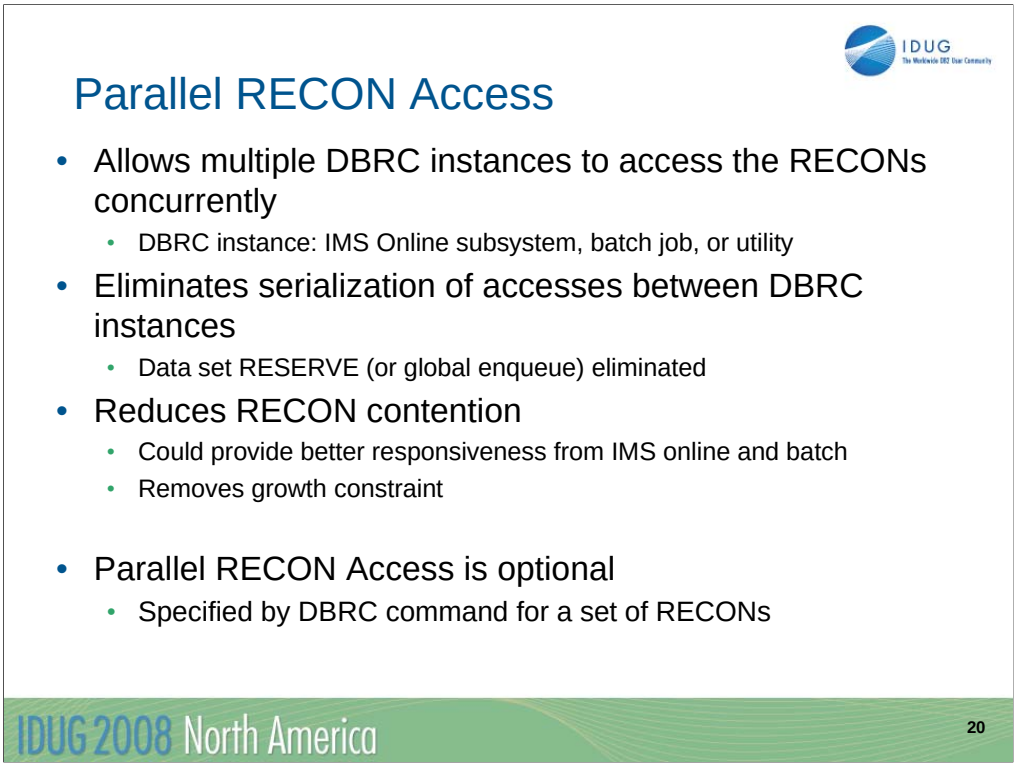
Updates to the RECONs may be done by using the interface to issue the INIT, CHANGE, and NOTIF DBRC commands.

The QUERY command has been enhanced to do queries directly for database data sets and partitions without querying the database. Queries for logs may be done for a range of log records by including a "from time" and/or a "to time".

Multiple sets of RECONs and ACBLIBs may be processed more easily through the specification of alternate DD names for these data sets. In V9 only RECON1, RECON2, RECON3, and IMS could be used for these DD names. This meant that programs which needed to access multiple RECONs were forced to dynamically allocate (SVC 99) alternate RECON data sets. In IMS V10 alternate RECONs may be specified using alternate DD names. Since some DBRC commands access ACBLIB, a similar facility has been added for an alternate DD name for ACBLIB. This allows users to include multiple RECONs and ACBLIBs in their DD statements.

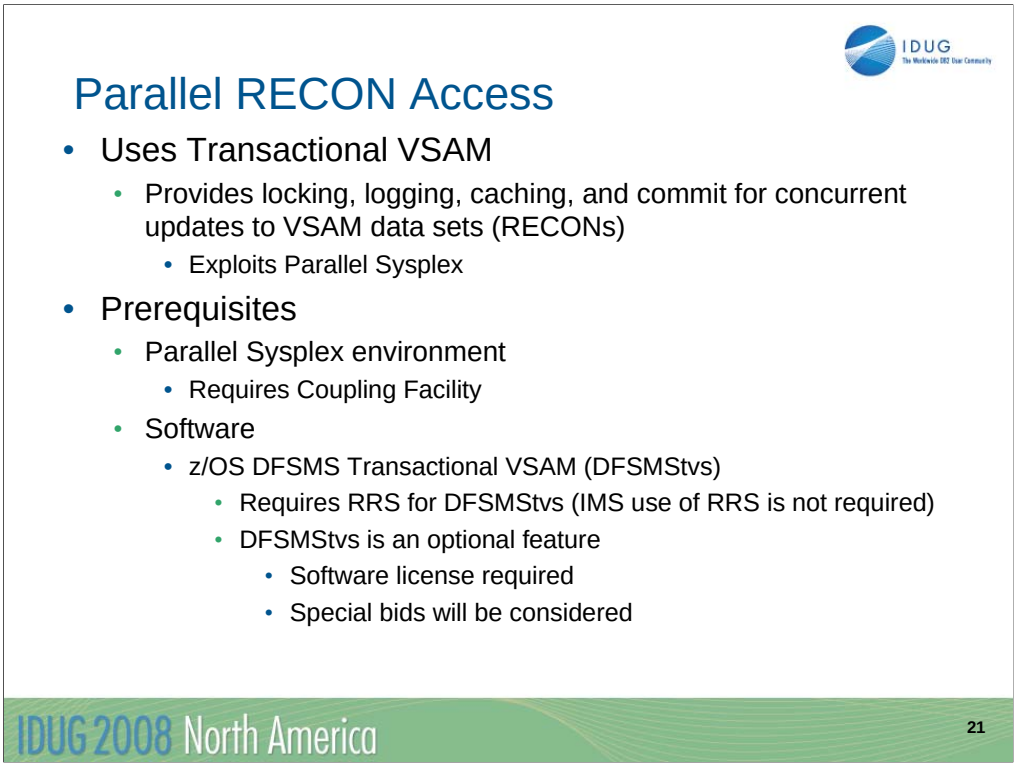
IMS V10 allows DBRC API programs to register as subsystems and authorize databases, partitions, and areas. This can be used by DBRC API programs which will provide utility functions such as image copies, reorganizations, and recoveries.

SAF security, such as RACF, is now invoked by the DBRC API. Queries are treated like LIST commands. This protects DBRC API commands and queries using the same facility that is used for DBRC commands.



Parallel RECON Access (PRA) is a new optional capability in IMS V10. It allows multiple DBRCs to access the RECONs at the same time.

Without PRA each DBRC instance must take its turn in accessing the RECONs. Multiple DBRCs may have the RECONs open at the same time, but only one may do I/O to the RECONs at any time. This restriction is eliminated with PRA. Serial access uses RESERVEs or global enqueues to serialize access to the RECONs. PRA eliminates this serialization. This reduces contention for the RECONs. It could provide better responsiveness, especially in situations where multiple online, batch, or utility executions of IMS are doing many I/Os to the RECONs.



PRA uses Transactional VSAM. This is a system facility that is provided by DFSMS. Transactional VSAM provides locking, logging, caching, and commit coordination for VSAM data sets such as the RECONs. It requires and exploits Parallel Sysplex.

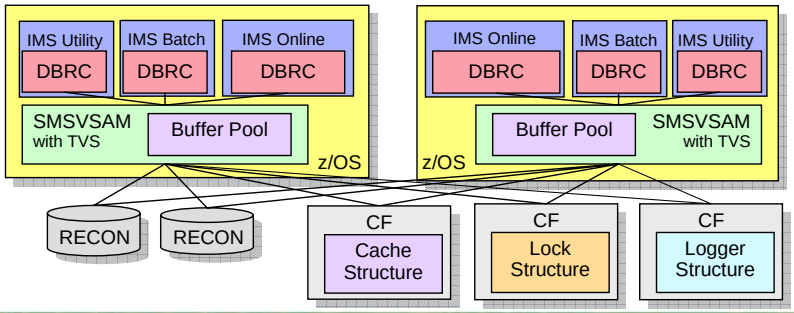
PRA requires a Parallel Sysplex including a Coupling Facility. This is true even when all of the DBRC instances are running in only one z/OS system.

PRA requires DFSMStvs. This is an optional feature of DFSMS. There is a licensing fee associated with this feature. Special bids will be considered for IMS customers using the Parallel RECON Access function, who do not already have DFSMStvs, to acquire DFSMStvs for use restricted to IMS.

DFSMStvs uses Resource Recovery Services (RRS). RRS is used for commit coordination. IMS use of RRS is not required. That is, the RRS=Y IMS execution parameter is not required.

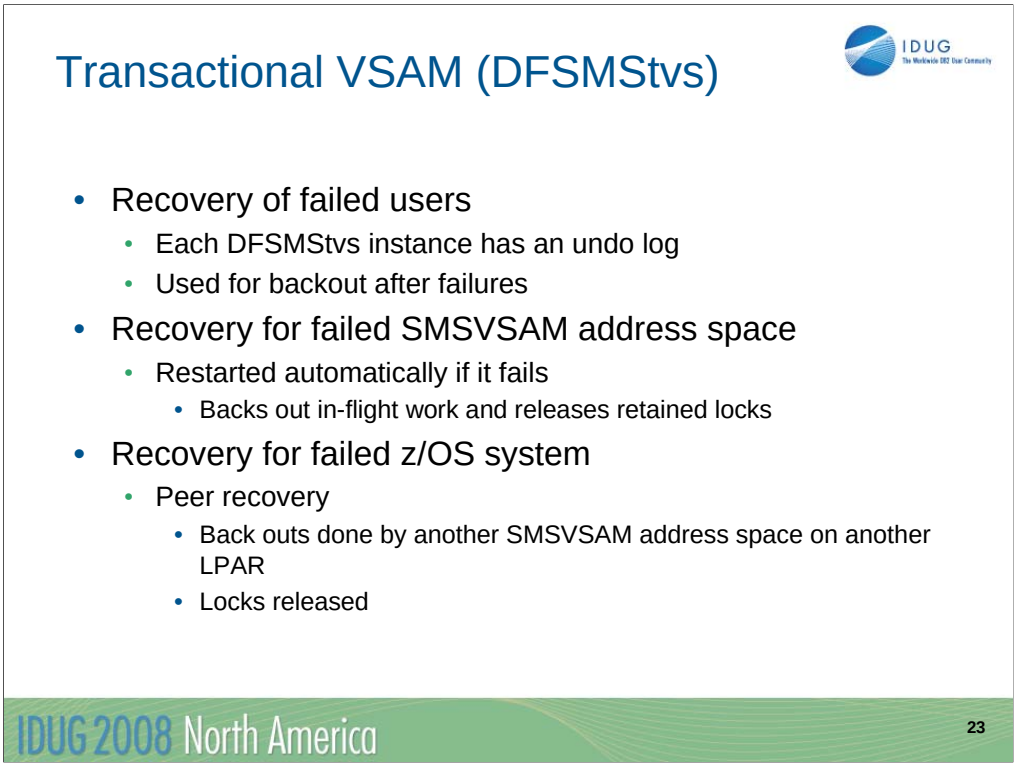
Transactional VSAM (DFSMStvs)

- TVS uses a cache structure in CF and a buffer pool in SMSVSAM address space
  - When a buffer in one SMSVSAM is updated, buffers with the same record in other SMSVSAM address spaces are invalidated
- VSAM record is locked when accessed by a user of TVS
  - SMSVSAM has its own lock manager
  - RECON record locked by a DBRC instance



This illustrates a DFSMStvs environment with DBRC users. When a DBRC instance accesses a VSAM record in the RECONs, the request is processed by TVS in the SMSVSAM address space. The SMSVSAM address space has its own buffers. The buffer pools are maintained in data spaces owned by the SMSVSAM address space. When using TVS, DBRC does not have its own buffers for RECON processing. Instead, it uses the SMSVSAM buffers. These buffers are shared with all users of the SMSVSAM address space. These are all of the VSAM data sets in the z/OS system which are using RLS or TVS. SMSVSAM also uses cache structures in the Coupling Facilities. Each shared data set is assigned to a structure. When a VSAM record in an SMSVSAM buffer pool is updated, if it resides in buffers in other systems those buffers are invalidated. This is done using the cross invalidation capability associated with cache structures and Parallel Sysplex.

The processing of VSAM requests includes lock requests. SMSVSAM has its own lock manager. It does not use the IRLM, however, its lock manager provides functions similar to those provided by IRLM for IMS and DB2 databases. The owner of a lock request is the DBRC instance. This is an IMS online system, an IMS batch job, or an IMS utility. Lock information is held in a lock structure in the CF.

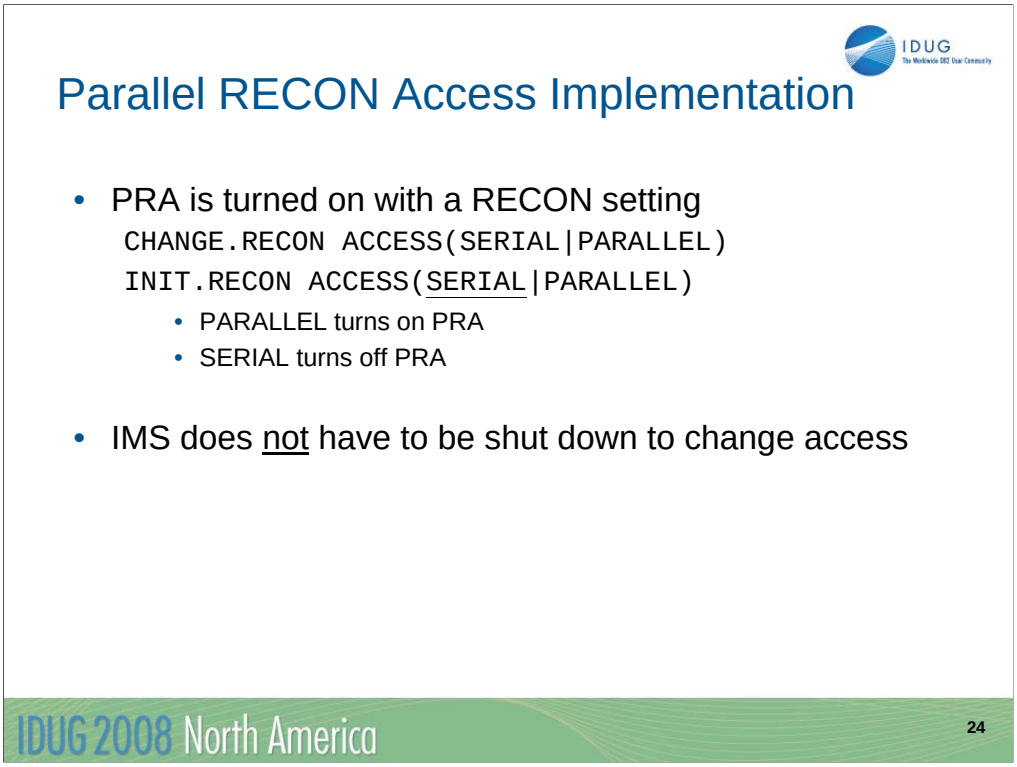


DFSMStvs has recovery capabilities to handle various kinds of failures.

Each DFSMStvs instance has its own undo log. These are log records which have the "before images" of records. If a user of DFSMStvs, such as DBRC fails, its uncommitted updates are backed out by DFSMStvs. The undo log is used for this purpose.

If DFSMStvs fails or the SMSVSAM address space fails, it is automatically restarted. When it restarts is backs out any in-flight work and releases any locks held by the in-flight transactions.

If a z/OS system fails, peer recovery is invoked. Back outs are done for the failed DFSMStvs by using another SMSVSAM address space in the Parallel Sysplex. The locks held for the failed work are released.



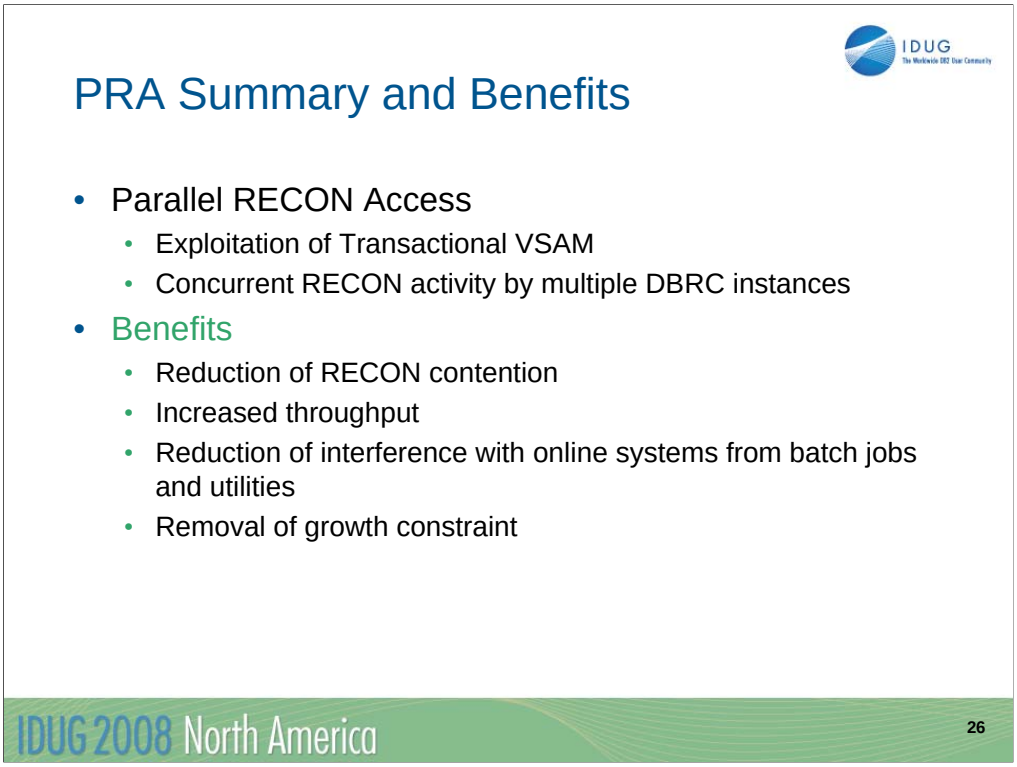
Parallel RECON Access is turned on by specifying ACCESS(PARALLEL) on the CHANGE.RECON or INIT.RECON command. It may be turned off with ACCESS(SERIAL). SERIAL is the default for the INIT.RECON command.

IMS does not have to be shut down to switch to parallel mode or to switch back to serial.

PRA Migration and Coexistence

- Parallel RECON Access cannot coexist with serial access for a set of RECONs
- PRA requires MINVERS('10.1')
- PRA requires DFSMStvs environment
  - IGDSMSxx parameters
  - Structure and log stream definitions
  - SHCDS data sets
  - RACF authority
  - RRS
  - SMSVSAM address space
  - Updated operation and recovery procedures

This is a summary of the migration and coexistence considerations.



PRA exploits Transactional VSAM for the RECON data sets. This allows multiple DBRC instances, which are online systems, IMS batch jobs, and IMS utilities, to access the RECON data sets concurrently. Volume or global data set serialization is not required. This reduces RECON contention. The reduced contention provides for greater growth potential and increased throughput possibilities. The primary recipient of the benefits for most installations will be online systems since they will suffer less interference from batch jobs and utilities which are accessing the RECONs.

IMS Version 10

Integration
Manageability
Scalability

- Database Enhancements
  - HALDB ILDS Rebuild Enhancement
  - Image Copy 2 Fast Replication
  - Fuzzy User Image Copy Support
  - ACBGEN Exploitation of Storage Above 16M
  - Fast Path Command Enhancements
  - Fast Path Capacity Enhancements
- DBRC Enhancements
  - DBRC Timestamp Precision
  - RECON READONLY Access
  - Improved SAF Support for RECONs
  - DBRC API Enhancements
  - Parallel RECON Access

IMS V10 Database and DBRC Enhancements

Rich Lewis
IBM
rzlewis@us.ibm.com