



B84

Understanding Fast Path DEDB Performance

Bill Stillwell

IMS Advanced Technical Support

IMS
Technical Conference

Sept. 27-30, 2004

Orlando, FL



Audience

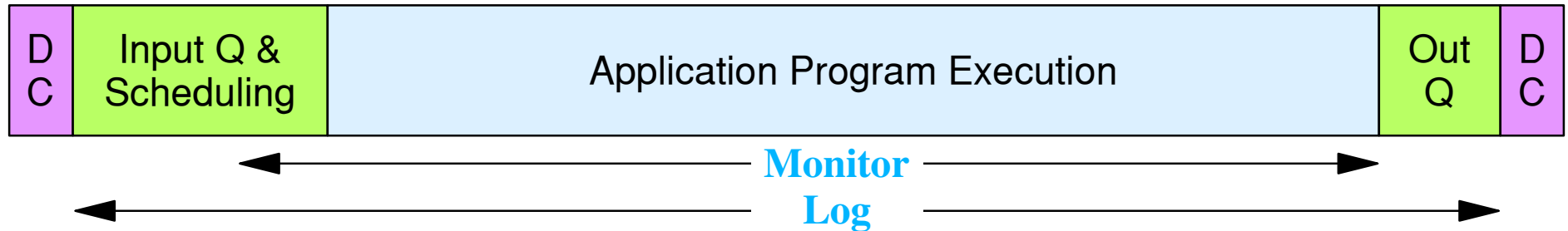
- ★ Users familiar with DEDB Fundamentals

Topics

- ★ Response time components
- ★ DEDB factors
- ★ Reports



Performance Data Collection



IMS log - primary purpose is recovery

- ❑ Begins logging when transaction arrives and is queued
- ❑ Ends logging when transaction sent and dequeued
- ❑ Also captures some schedule level and system level utilization statistics
 - # and type of calls, buffers, logging, pool utilization, failures, ...

IMS monitor - primary purpose is analysis

- ❑ More detailed recording of IMS activities during scheduling and application program execution
 - Occurrences and elapsed times, I/Os, lock and latch contention, ...
- ❑ Also captures utilization statistics

Performance Reporting

IMS log-based reporting

- IMS utility
 - FP Log Analysis Utility (DBFULTA0)
- Tools
 - IMS Performance Analyzer (5655-E15)

IMS monitor based reporting

- IMS utility
 - IMS Monitor Print Program (DFSUTR20)
 - Note: DFSUTR20 does not report fast path activity
- Tools
 - IMS Performance Analyzer (5655-E15)

Database Analysis

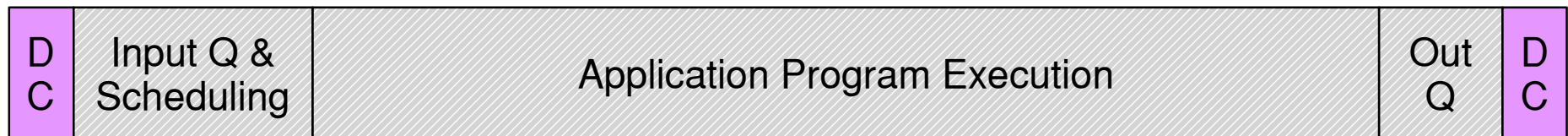
- FP Basic and Online Tools (5655-E30 and 5655-F78)
 - DEDB Pointer Checker (Basic and Online)
 - DEDB Tuning Aid (Basic)

DC Component

Time spent in the network for input and output messages

- SNA
 - Network, VTAM, (APPC/MVS), IMS DC → message queue
- TCP/IP
 - Network, TCP/IP, Server, OTMA (DC) → message queue
- *Independent of whether transaction uses Fast Path (EMH) or Full Function queuing and scheduling*

DC component not captured by IMS logging or IMS monitor



Input Queue

Full function messages

- ❑ Stored in QPOOL in control region
- ❑ Queued on SMB (transaction control block)
 - Associated with transaction code
- ❑ May be written to message queue data sets

Fast path messages

- ❑ Stored in EMH buffer (dedicated to input terminal)
 - FP transactions must be single segment, response mode, non-conversational, non-MSB, ...
- ❑ Queued on balancing group (BALG)
 - Associated with scheduled PSB in one or more IFP region(s)
 - Transaction rejected if PSB not already scheduled
 - Transaction discarded if all scheduled PSBs terminate
- ❑ Never written to message queue data sets

Scheduling

Full function scheduling

- ❑ Scheduled into *MSG* region
 - Stays queued if no region available
- ❑ Many factors affect scheduling
 - Scheduling priorities, classes, limit counts, ...
 - Quick reschedule
 - Transaction can be *WFI*
 - *MSG* region can be *PWFI*



Fast path EMH scheduling

- ❑ Scheduled in *IFP* region
 - Not queued if no eligible region started (no active *BALG*)
- ❑ No scheduling parameters
 - *FIFO*
- ❑ *IFP region* is *WFI*

EMH scheduling is most significant when application is not complex.

Output Queue and DC

Both EMH and Full Function

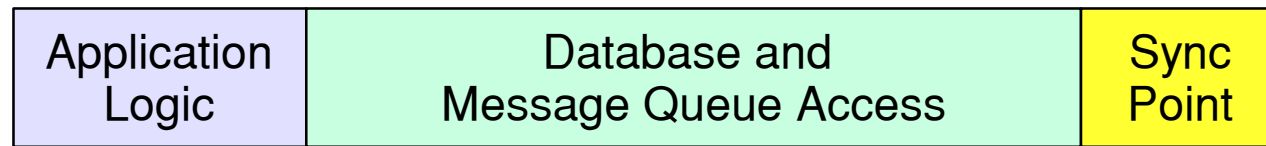
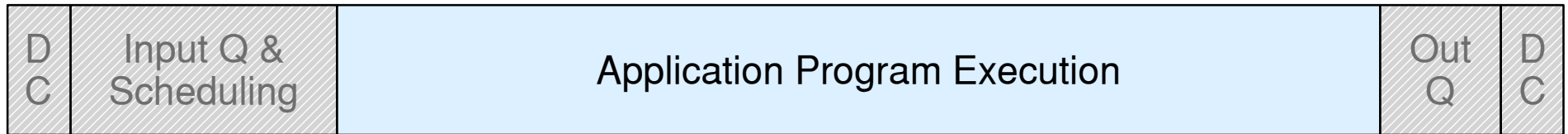
- Wait for commit (sync) record to be physically written before sending response
 - Response message in QPOOL (FF) or EMH buffer (FP)
 - Sync point log record not forced to WADS or OLDS
 - Waits for ...
 - OLDS buffer to fill, or ...
 - Another process to "checkwrite" buffers to WADS, or ...
 - Timer pop



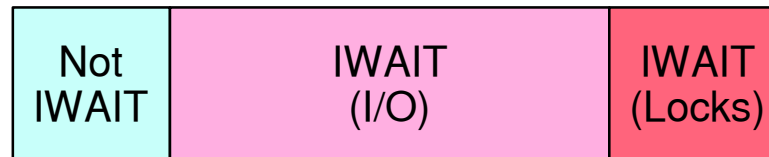
- Similar performance characteristics

The rest of this presentation addresses only the application program execution component

Application Program Execution



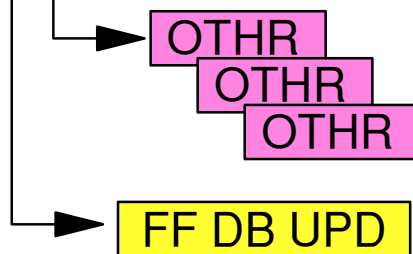
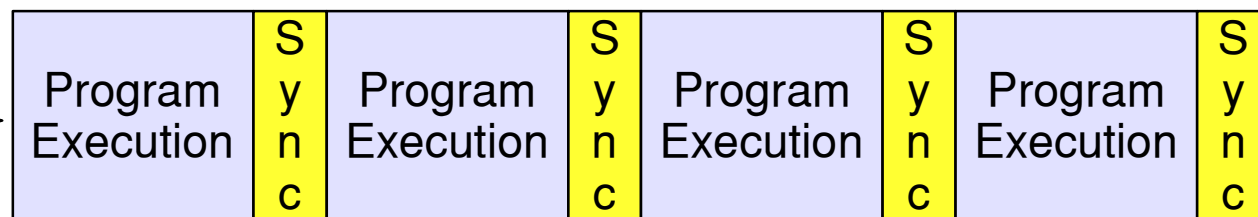
Components of a DL/I Call →



Series of DL/I Calls →



Series of Transactions →



Asynchronous parallel processes
(DEDB Output Threads)

Synchronous parallel/serial processes
(OSAM/VSAM)

Application Performance Components

Once scheduled in dependent region
.... each transaction has same performance characteristics

- No difference between execution in MSG and IFP regions
 - Only queuing and scheduling are different
- *DEDBs have same performance characteristics in MSG, BMP, and IFP regions*

Application Performance ...

Factors affecting "elapsed execution" time in dependent region to process a transaction

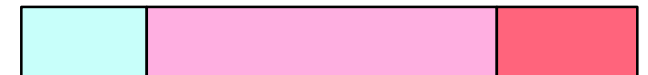
□ Application

- Dependent region initialization - load program
- Execute application logic (CPU)
- Wait for system resources (CPU, paging, ...)
- Perform non-IMS activities (DB2, MQ, ...)
- Perform DL/I activities



□ DL/I activities

- TM calls - get and insert FF or FP EMH messages
- DB calls - fast path DEDB and full function databases
- System calls
 - INQY, INIT, LOG, SETO, ...
 - GU IO-PCB, CHKP, or SYNC
- Sync point processing

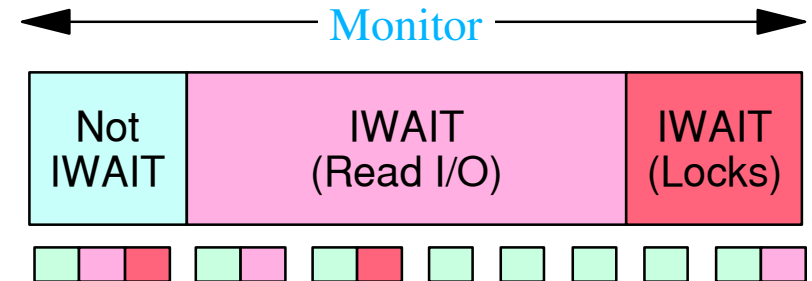


DEDB Call Processing

Factors ...

- Application
- DEDB calls
 - Not I-Waiting - for example
 - Executing **IMS code**
 - Waiting for **system resources** (CPU, paging, MONITOR, ...)

- I-Waiting
 - Waiting for **DB I/O**
 - Waiting for **IMS resources** (locks, latches, buffers, ...)



In this context, an I-Wait is a wait captured by the IMS Monitor. There may or may not be an IMS IWAIT issued.

- Sync point processing

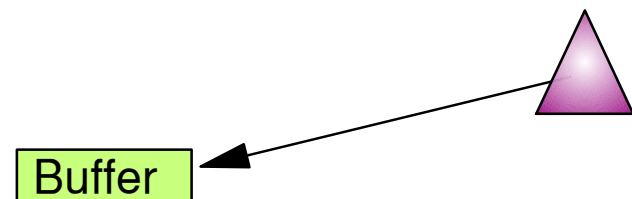
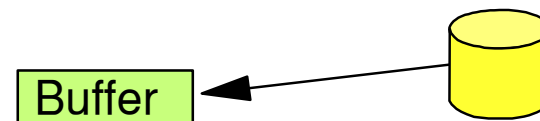
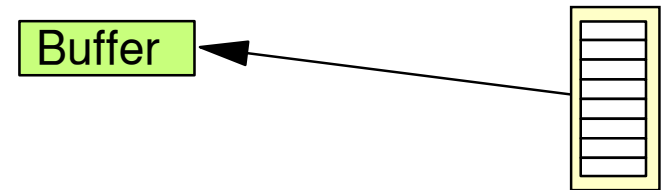
DEDB Call ...

Not I-waiting

- Call pathlength (CPU and system waits)
 - Includes VSO GET from data space

I-waiting for ...

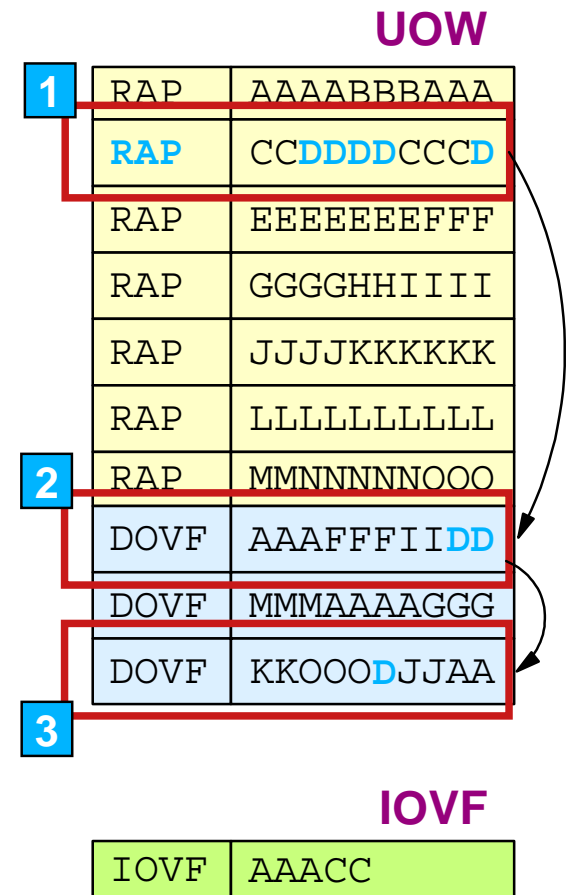
- IMS resource
 - Locks on DB resources
 - CI, UOW, Area, ...
 - Latches to serialize processes
 - OBA, SDEP commit, ...
 - Buffers for read I/Os
 - Available buffer (all buffers are in use)
- Read I/O from DASD
 - To retrieve segment
 - To find space to insert segment
- VSO GET from CF



DEDB Call ...

Get call (e.g., get all segments in record 'D')

- ❑ Call randomizer (Area/RAP) and/or follow pointer (RBA) to determine CI
- ❑ Look for CI in **local** buffers
 - No look-aside buffering for DEDBs
- ❑ If not in local buffer pool
 - **Get CI lock** (may have to iwait)
 - **Get buffer** (may have to iwait)
 - **Read CI from DASD** (iwait)
or **VSO** (not-iwait)
- ❑ Look for segment in **CI**
- ❑ If not in CI
 - Follow pointers until found (or 'GE')
 - May have to **repeat**
"get lock, get buffer, read CI" steps



DEDB Call ...

Get call ...

- When segment found
 - Expand if segment compressed
 - Pass segment to program I/O area

On next Get call (e.g., Get 'E')

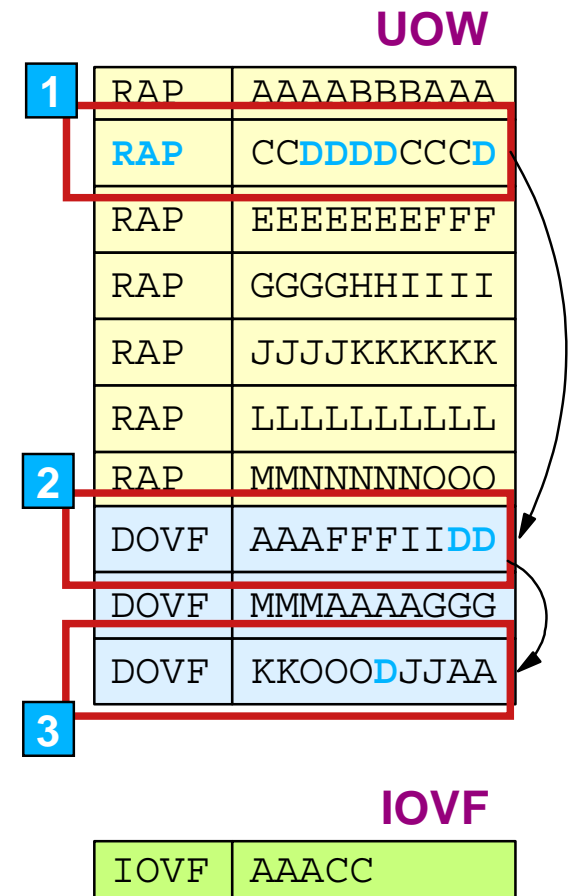
- Locks no longer needed (CIs containing record 'D') are not released

Delete call

- Delete data in buffers
 - May require additional CIs to retrieve and delete children or update pointers

GHU ROOTD (1 I/O)

DLET (2 more I/Os)

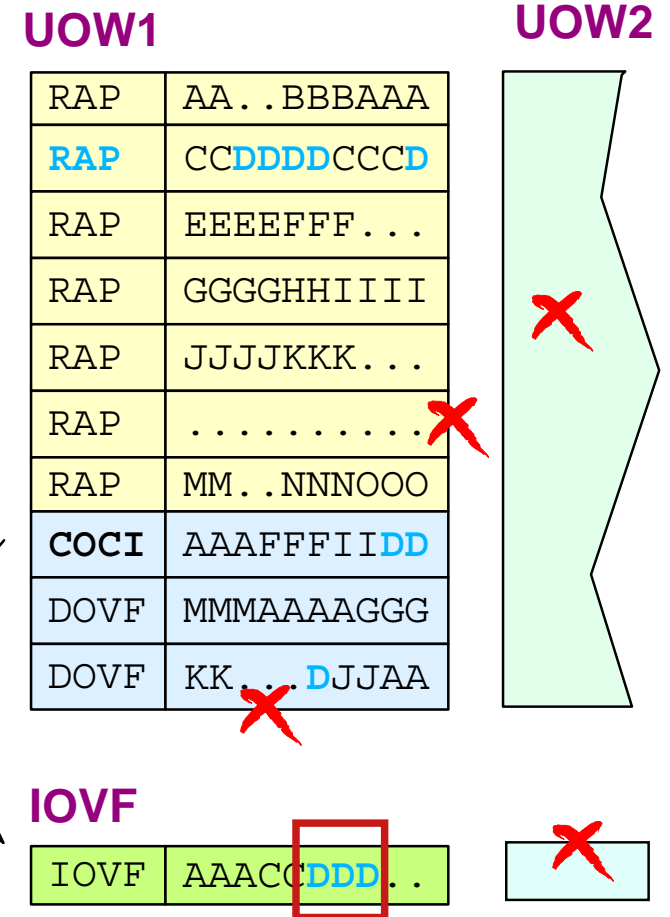


Note: Updated CIs are not written to DASD until after sync point completes and logs written to DASD.

DEDB Call ...

Insert call

- Establish position for insert
 - Similar to retrieval call
- Determine "most desirable CI"
 - For root segment - RAP CI
 - For dependent segment - Root CI
- Try to place segment in MDCI
- If no room, find space using COCI pointer (probably involves *locking and I/O*)
 - In DOVF (always look here first)
 - In IOVF (if no room in DOVF)
 - Never in another RAP CI or another UOW
 - Never share an IOVF CI with data from another UOW



DEDB Call ...

Replace call

- If segment length does not change
 - Replace in place
 - May be impacted by **compression**

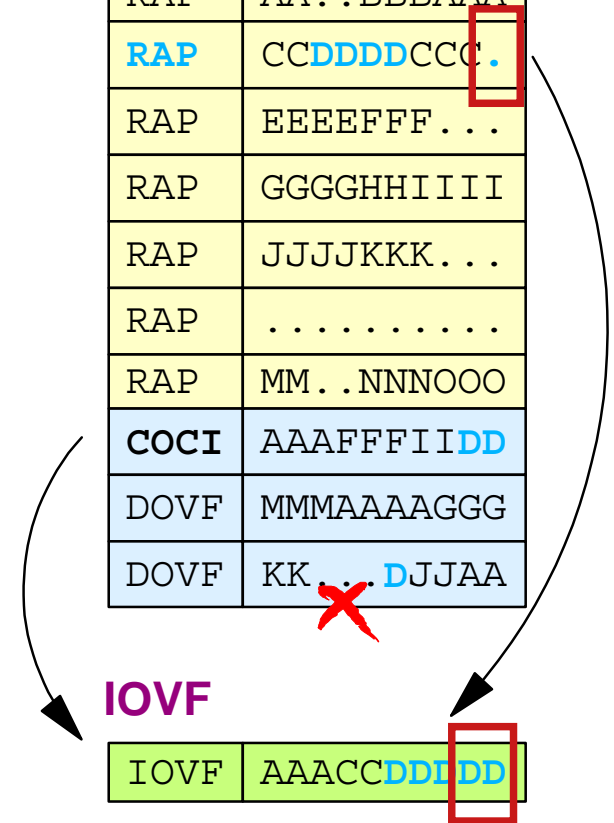
- If segment length changes (larger or smaller)
 - Internal delete and insert
 - Try to place in MDCI
 - Segment may move
 - Could be more I/O
 - "get lock, get buffer, read CI"
 - If using segment compression, should consider specifying minimum length

UOW1

RAP	AA..BBBAAA
RAP	CC DDDD CCC.
RAP	EEEEFFF...
RAP	GGGGHHIIII
RAP	JJJJKKK...
RAP
RAP	MM..NNNOOO
COCI	AAAFFFI IDD
DOVF	MMMAAAAGGG
DOVF	KK... DJJAA

IOVF

IOVF	AAACC DDI DD
-------------	---------------------



Buffer Management during Call

Allocation buffer for read I/O (or VSO get)

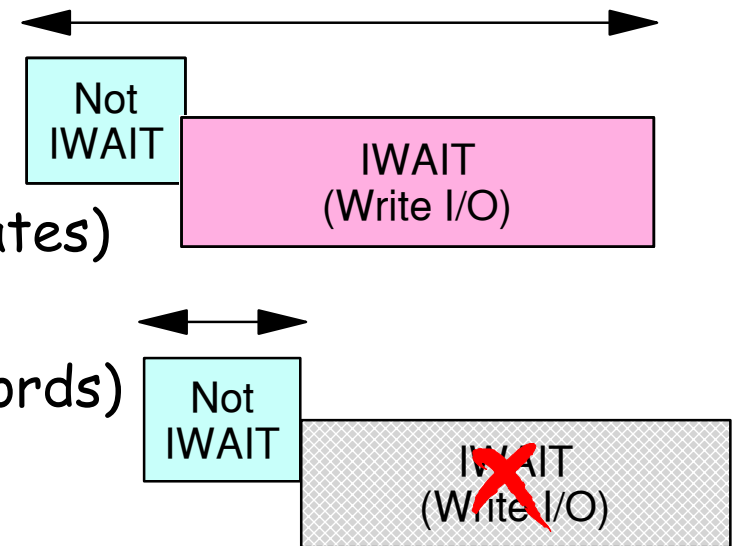
- If all NBA buffers in use
 - *Steal* unmodified buffers for reuse
 - Release locks on stolen buffers if no longer needed
- If cannot steal from NBA
 - Get OBA latch to use OBA buffers
 - May have to wait for latch (reported as **OB CONTENTION**)
 - No problem if don't have to wait
- Allocate buffer from available buffers
 - /DIS POOL FPDB
 - FPDB POOL:**
 - AVAIL=1846 WRITING=92 PGMUSE=419 UNFIXED=2643**
 - If none AVAIL, program waits (reported as **BW CONTENTION**)
- Program cannot exceed NBA + OBA
 - **U1033abend** (or '**FR**' status for BMP)

Application Performance (Sync)

Factors ...

- ❑ Application
- ❑ IMS database calls

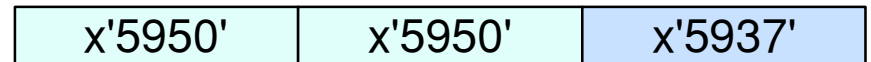
- ❑ Sync point processing
 - Full function database
 - *Physical logging (I/O)*
 - *Database I/O* (write committed updates)
 - Fast path
 - *Logical logging* (create all FP log records)
 - May have to I-wait for LOGL latch
 - Database updates written later
 - Commit record
 - Full function (x'37') and fast path (x'5937')
 - *Logical logging only (except BMP)* - output message processing waits for physical logging to occur asynchronously



DEDB During Sync Point

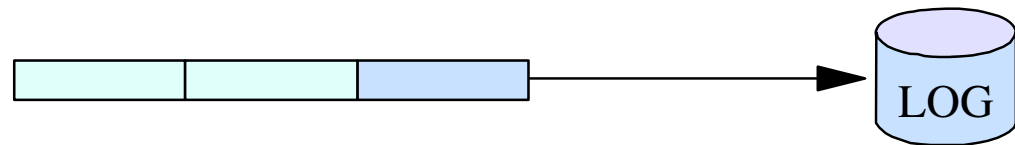
Phase 1 (before commit)

- Logical logging of all fast path log records
 - Get LOGL latch

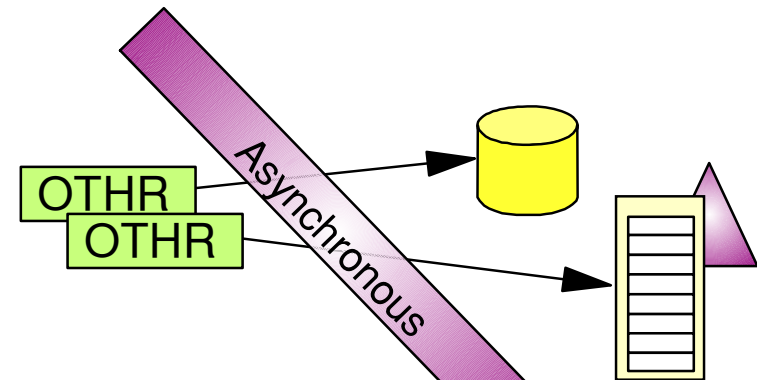


Phase 2 (updates are committed)

- Physical logging
 - OLDS or WADS



- Output thread processing
 - DASD writes
 - VSO PUTs
- Locks and buffers held
 - Until output thread completes



Transaction response does not wait for output threads to complete.

Sync Point Processing

Phase 1

- ❑ Logically log all updates
 - x'5950' (and sometimes a few others)
- ❑ Logically log x'5937' sync record
 - Equivalent to full function x'37'
- ❑ May want more OLDS buffers to handle sudden spikes during sync point processing
 - Especially if high update BMPs
 - Monitor logger statistics in x'4507' log record

Physical logging

- ❑ BMPs
 - Checkwrite x'5937'
- ❑ MPPs and IFPs
 - Wait for buffer to be written when buffer full, or ...

Sync Point Processing ...

Phase 2

- Release all unneeded buffers and locks
 - Buffers/CIs containing unmodified data
- Transfer lock ownership of updated CIs to Output Threads
 - Once transferred, even this program cannot access CI

GU ROOT1

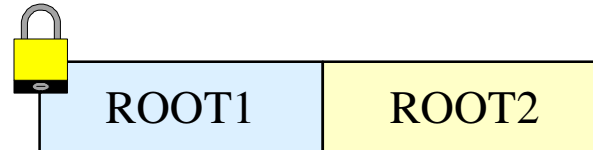
Lock CI

SYNC

Transfer lock to output thread

GU ROOT2

Wait for lock (CI contention)



- Happens most frequently in sequential BMPs
 - Can be avoided by using HSSP

Output Thread Processing

Invoked by physical logger

- When x'5937' written to OLDS or WADS

Output Threads

- One per area - in parallel if multiple areas
- Execute under SRB in control region
- Updated CIs chain-written to Area Data Sets
 - Asynchronous to dependent region processing
- Seldom a problem unless not enough threads (OTHR=nnn)
 - Very cheap - be generous - set OTHR=255
- When complete
 - Buffers freed and locks released

Transaction response

- Does not wait for output threads to complete

DEDB vs Full Function - FF Locking

Record locks

- ❑ RAP, root RBA, or root key
 - Sometimes individual segments are locked
- ❑ More frequent lock requests (usually)
- ❑ Other records in same block/CI available to other programs
 - Unless on same RAP chain
- ❑ Lock released when position changes
 - Unless segment updated

DEDB vs Full Function - FP Locking

CI locks - for non-HSSP processing

- ❑ Fewer lock requests but (usually) larger scope
 - Can't share CI with other transactions
- ❑ Locks not released until sync point
 - Unless buffer stolen or DEQ call

UOW locks - HSSP BMP and Online Reorg

- ❑ Entire UOW locked - EXCL - even if PROCOPT=HG
 - Could be a lot of data
- ❑ Forces "transactions" to get UOW lock - READ - before getting CI lock

Area locks

- ❑ Usually for sequential dependent (SDEP) CI management

DEDB vs Full Function - Data Sharing

If data sharing FF database

- ❑ Requires OSAM/VSAM cache structures in CF
 - Accessed for every read (register)
- ❑ If block or CI updated
 - Get block lock from IRLM
 - May require buffer invalidation

If data sharing DEDBs

- ❑ Must use IRLM for all locking
- ❑ No block locks needed (already have CI lock)
- ❑ Non-VSO DEDBs
 - No cache structure required - No buffer invalidation needed
 - Can't be in another IMS's buffer
- ❑ Shared VSO
 - Requires cache structure, registration, buffer invalidation

DEDDB vs Full Function - FF I/O

FF *read and write* I/O is synchronous

- Wait for all read I/O (obviously)
 - But may be fewer reads because of buffer look-aside

- Wait for write I/O
 - Buffer stealing
 - One buffer at a time
 - Sync point processing
 - VSAM - one CI at a time
 - OSAM - parallel chained writes


- OSAM sequential buffering
 - Provides *asynchronous chained* read-ahead capability for sequential processes

DEDB vs Full Function - FP I/O

Fast Path read I/O is synchronous

- Wait for CI to be read into buffers (again, obvious)
 - May be more reads since no buffer look-aside for DEDBs

FP write I/O is asynchronous

- Updated CIs written asynchronously after sync point by *output threads*
 - Waits for physical logging of sync record
 - Chained / parallel writes
- *Response and dependent region don't wait* for output threads to complete 

HSSP processing

- Addressed later

DEDB vs Full Function - Buffer Mgmt

Full function buffer management

- FF supports buffer look-aside
 - Can share/reuse buffers by different transactions
- Never run out of buffers
 - May require buffer stealing

Fast path buffer management

- FP does not support buffer look-aside (except for S-VSO)
 - Buffer freed (and forgotten) after sync point processing
 - Limits reuse of buffer by different transactions
- May run out of buffers for dependent region
 - Must allocate enough buffers (NBA+OBA)
 - May steal unmodified buffers
 - May wait for overflow buffer (OBA) latch (easy to avoid)
 - May wait for available buffer (should be rare - easily fixed)

DEDB vs Full Function - Logging

FF logging

- FF logs as it goes
 - Logical logger called for every update
 - Requires logical logger latch (LOGL)
- During sync point processing
 - Forces (checkwrites) DB update logs to DASD during Phase 1

FP logging

- Updates logged only during sync point processing
 - Logical logger called once with "list" of log records
 - Requires logical logger latch
 - May log more than necessary if LGNR too small (easily fixed)
- Physical logging not forced (except for BMPs)
 - Output threads wait for physical logging to occur

What to Look For

Look for things you can do something about

- ❑ Locks
 - Number
 - Elapsed time
- ❑ Read I/O
 - Number
 - Elapsed time
- ❑ Buffer management
 - Wait for OBA latch
 - Wait for buffer
- ❑ Logging
 - Combining constant (LGNR exceeded)
- ❑ Shared VSO
 - Look aside buffering
 - CF service times

Where (and where not) to Look

Database Analysis

- FP Basic and Online Tools (5655-E30 and 5655-F78)
 - DEDB Pointer Checker (Basic and Online) ✨
 - DEDB Tuning Aid (Basic)

IMS log-based reporting

- IMS utility
 - FP Log Analysis Utility (DBFULTA0)
- Tools
 - IMS Performance Analyzer (5655-E15) ✨

IMS monitor based reporting

- IMS utility
 - IMS Monitor Print Program (DFSUTR20)
 - Note: DFSUTR20 does not report fast path activity
- Tools
 - IMS Performance Analyzer (5655-E15) ✨

Where to Look

Fast Path Basic Tools - DEDB Pointer Checker

- Reads database (area) and reports on database (dis)organization
- Reports
 - Free Space Analysis
 - Record and Segment Occurrence Analysis
 - Record and *Segment Placement Analysis*
 - *Record and Segment I/O Analysis*
 - Root Distribution and Synonym Chain Analysis
 - UOW Reports

DEDB Pointer Checker

"Predicts" what I/O performance will be

- "Segment Placement Analysis" and "Segment I/O Analysis"

SEGMENT PLACEMENT ANALYSIS

SEGNAME	SCD	LVL	TOT #OCCS	---IN RAA BASE---		----IN DOVF----		----IN IOVF----	
				NO. OCCS	P/C	NO. OCCS	P/C	NO. OCCS	P/C
TSSROOT	1	1	83	44	53.0	17	20.5	22	26.5
TSSDIR1	3	2	317	189	59.6	76	24.0	52	16.4
TSSD11	4	3	6	0	0.0	6	100.0	0	0.0
TSSD111	5	4	21	0	0.0	21	100.0	0	0.0
TSSD12	6	3	0						
TSSDIR2	7	2	676	225	33.3	261	38.6	190	28.1
TSSDIR3	8	2	173	93	53.8	39	22.5	41	23

SEGMENT I/O ANALYSIS

.....

Look here first

Not good!!

Look at "Root Distribution and Synonym Chain Analysis"

** RECORD I/O **	AVG:	2.58	SDEV:	0.46	MAX:	6	MIN:	1
*** ROOT I/O ***	AVG:	1.47	SDEV:	0.25	MAX:	3	MIN:	1

Where to Look ...

Fast Path Log Analysis (DBFULTA0)

- Reads IMS log and reports only fast path activity based on content of log records
 - Does not report any DEDB elapsed time

- Reports
 - Detail Listing of Exception Transactions
 - Overall Summary of Resource Usage and Contentions
 - Summary of VSO Activity

IMS Performance Analyzer has similar log-based reports.

Where to Look ...

IMS Performance Analyzer - Log Based Reports

- FP Transit reports
 - EMH Transit Time Analysis
 - EMH Transit Log
 - *FP Transaction Exception Log* - includes IFP/MPP/BMP FP activity but transit queue time only reported for IFP regions
- FP Resource Usage reports
 - FP Resource Usage and Contention
 - FP DB Call Statistics
 - DEDB Update Activity
 - DEDB Update Trace
 - VSO Statistics

IMS PA - Log-based Reports

"Fast Path Transaction Exception Log"

- Up to 4 lines for each transaction
 - Can use lots of paper
 - "Transit Times" not reported for non-EMH transactions

Fast Path (EMH) Transaction Exception Log

Log 10Jun2001 19.01.37.44

Report 14.50 21Jun2001

Sync Point Time	S F	Trans Code	Routing Code	P T	User ID	PST ID	Queue Count	-Transit Times (Msec)-				Output (sec)	-DB Call-		--ADS--		--VSO--		Buf Use	--DB Wait--			
								In-Q	Proc	Out-Q	Total		DEDB	MSDB	Get	Put	Get	Put		CI	UW	OB	CB
19.01.37.45		ORDER	ORDER		Jane	2	7	76	132	43	251	1.56	146	0	57	14	7	2	24	5	2	1	2
DEDB Calls - GU= 12 GN= 32 GNP= 0 GHU= 57 GHN= 32 GHNP= 0 REPL= 6 ISRT= 2 DLET= 5 FLD= 0 POS= 0 TOTAL=146 Buffer - NBA= 20 OVFN= 4 STEAL= 2 WAIT= 2 OTHR= 14 NRDB= 0 PBUF= 0 PBWT= 0 ASIO= 0 AIOW= 0 VSO - VGET= 7 VPUT= 2 DGET= 0																							

IMS PA - Log-based Reports ...

"FP Resource Usage and Contention Report"

- Summarizes by transaction code
 - Routing code not include if not IFP (*MPP)

Fast Path Resource Usage and Contention - IMSA

From 10Jun2001 19.01.37.44 To 10Jun2001 19.03.43.47 Elapsed= 0 Hrs 2 Mins 06.029.096 Secs

Trans Code	Routing Code	Count	---DEDB Calls---				--- ADS I/O ---				--VSO Activity--				-Common Buffer- Usage				Contentions		LGNR	Stat	Totl	Tran	
			Reads	Updates	Reads	Updates	Reads	Updates	Reads	Updates	Avg	Max	Wts	Stl	UOW	OBA	CI/Sec	Total #CI	Sync	Rate					
BILL	*MPP	2904	12	16	3	3	2	2	2	2	2	5	2	2	4	7	0	0	23	5	17	10	4	0	23
ORDER	ORDER	18381	6	8	2	3	1	1	1	1	1	1	1	1	1	0	0	154	7	13	16	8	0	145	
PARTS	*BMP	7128	37	39	16	18	4	6	7	11	7	12	16	22	2	2	1	1	79	2	18	4	2	0	56
STOCK	STOCK	280	5	5	4	4	1	2	2	3	2	4	3	3	1	1	0	0	12	14	22	1	1	0	2

- Notice STOCK transaction
 - 280 trans/126 sec = 2.2 tps
 - 22 CI contentions per second / 2.2 tps = 10 CI contentions/transaction

IMS PA - Log-based Reports ...

"DEDB Update Activity Report"

- ❑ Number of DDEP and SDEP segments updated
- ❑ Segment update rate
- ❑ ...

DEDB Update Activity - IMSA

From 14Apr2001 8.54.43.80 To 14Apr2001 8.54.46.00 Elapsed= 0 Hrs 0 Mins 2.201.638 Secs

Database Name	Area Name	Root/DDEP Update	SDEP Insert	ADS Open	New EQE	Updates /sec	--- First Update Date	---- Update Time	--- Last Update Date	----- Update Time .
ACCOUNTS	ACCOUNTS	9	12	0	0	16.2	14Apr2001	8.54.44.57	14Apr2001	8.54.45.86
CLIENTS	CLIENTA1	6	6	0	0	6.1	14Apr2001	8.54.43.86	14Apr2001	8.54.45.80
	CLIENTA2	2	2	0	0	6.8	14Apr2001	8.54.44.62	14Apr2001	8.54.45.21
CLIENTS	*Total*	8	8	0	0	8.2	14Apr2001	8.54.43.86	14Apr2001	8.54.45.80
FINANCE	FINANCE	103	0	1	0	49.9	14Apr2001	8.54.43.94	14Apr2001	8.54.46.00
ORDERS	ORDERS	27	74	0	0	48.9	14Apr2001	8.54.43.94	14Apr2001	8.54.46.00
STOCK	STOCKA1	1	21	0	0	22.0	14Apr2001	8.54.45.39	14Apr2001	8.54.45.39
	STOCKA2	2	43	0	0	154.1	14Apr2001	8.54.44.69	14Apr2001	8.54.45.21
	STOCKA3	2	46	0	0	78.9	14Apr2001	8.54.45.35	14Apr2001	8.54.45.96
	STOCKA4	1	22	0	0	23.0	14Apr2001	8.54.45.43	14Apr2001	8.54.45.43
STOCK	*Total*	6	132	0	0	160.5	14Apr2001	8.54.44.69	14Apr2001	8.54.45.96
System Totals		4804	721	1	0	2509.4	14Apr2001	8.54.43.80	14Apr2001	8.54.46.00

Where to Look ...

IMS PA - Monitor-based Reports

- Reports
 - DEDB Resource Contention
 - Fast Path Buffer Statistics
 - OTHREAD Analysis
 - VSO Summary
 - ...

- Uses fast path activity captured by the IMS monitor
 - Requires at least IMS V7
 - Records "occurrences" and "elapsed times"
 - Log reports do not capture elapsed times

IMS PA - Monitor-based Reports

DEDB Resource Contention

- Summary information about locks and latches

Fast Path DEDB Resource Contention Summary

**** CI Lock IWAIT ****

Area Name	Sharing Type	Counts	Elap/Count Sc.Mil.Mic	StDev	Max IWAIT Sc.Mil.Mic	Pct Tot Counts	Pct Tot IW Elp
DB23AR0	A	3	3.313	0.466	5.498	9.09%	0.05%
DB23AR3	A	1	4.871.974	0.000	4.871.974	3.03%	24.50%
DD01AR0	A	13	3.880	0.499	6.863	39.39%	0.25%

Sharing Types:

A : Area / Non Level Share
 B : 1 IRLM Block Level Share
 C : 2 IRLM Block Level Share

**** Area Lock IWAIT ****

Area Name	Sharing Type	Counts	Elap/Count Sc.Mil.Mic	StDev	Max IWAIT Sc.Mil.Mic	Pct Tot Counts	Pct Tot IW Elp
BANKC00	C	11	18.813	0.129	22.795	39.29%	15.18%
BANKC01	C	17	68.036	2.828	837.022	60.71%	84.82%

Also reports on
 OBA, UOW, and
 other lock and latch
 waits.

Fast Path DEDB Lock Activity

Area Name	Shr Lvl	Loc Typ	Lock Count	Lock Elapsed				Pct Total Locks	Pct Total Lock El	IWAIT Count	IWAIT Elapsed			
				Average Sc.Mil.Mic	StDev	Maximum Sc.Mil.Mic	** Pct Total **				Average Sc.Mil.Mic	StDev	Maximum Sc.Mil.Mic	** Pct Total **
DB23AR0	1	CI	8	1.731.126	1.710	7.895.488	6.54%	8.87%	3	2.152	0.078	2.388	7.09%	0.04%
DD01AR0	2	CI	40	417.830	4.100	7.888.613	34.54%	10.71%	13	3.898	0.542	9.214	32.65%	0.28%

IMS PA - Monitor-based Reports ...

Buffer Statistics

- Reports on buffer utilization by region and total
 - Example shows "total" by transaction code

Fast Path Buffer Statistics

Region Totals		From 07Jun2001 11.25.11.80 To 07Jun2001 11.28.25.78										Elapsed= 0 Hrs 3 Mins 13.975.908 Secs					
Trans Code	No.of Sync	No.of Bufs Requested		No.of Bufs Updated		No.of Steal Invoc.		No.of Bufs Stolen		No.of Bufs Used (NBA)		No.of Bufs Used (OBA)		No.of WAITs for OBA		Elapsed Time for OBA	
		Avg	Max	Avg	Max	Avg	Max	Avg	Max	Avg	Max	Avg	Max	Avg	Max	Avg	Max
DDLTRN01	38	3	21	3	20	0	0	0	0	3	20	0	0	0	0	0.000	0.000
TXCDDS01	4	4	7	0	1	1	2	1	1	2	3	1	2	0	1	0.004	0.015

IMS PA - Monitor-based Reports ...

Output Thread Analysis

- Reports on output thread activities
 - Active, waiting, number buffers on queue

Fast Path OTHREAD Analysis

----- Active OTHREADS -----				----- Waiting Areas -----			----- Buffers on Queue -----		
Enq	Counts			Counts			Counts		
Counts	OTHREAD/Enq	StDev	Max value	Area/Enq	StDev	Max value	Buff/Enq	StDev	Max value
731	0.03	6.444	2	0.01	12.025	1	3.20	1.066	16

**** DEDB Write IWAIT ****

ADSname	Share	VSO	IWAITs	Elap/IWAIT	Max value	Pct Tot	Pct Tot	---- CI Write Count ----	
	Level			Sc.Mil.Mic	Sc.Mil.Mic	IWAITs	IWT Elp	CI/IWAIT	Max value
DB23AR0	1	NO	5	5.387	17.631	0.63%	1.89%	1	1
DB23AR1	1	NO	5	4.196	5.392	0.63%	0.38%	2	4
BANKC00	3	YES	426	4.404	140.204	47.53%	21.28%	2	8

IMS PA - Monitor-based Reports ...

VSO Summary

- Reports on all VSO activity for both shared and non-shared VSO areas

Fast Path DEDB VSO Summary

**** Preload ****

Area Name	Share Level	Start Time HH.MM.SS.TH	End Time HH.MM.SS.TH	Elapse Time Sc.Mil.Mic	No. of CI Read
BANKC00	1	15.03.09.85	15.03.09.90	52.019	150

**** I/O Activities **** (SHARELVL 0/1)

Area Name	Share Level	VSO Reads	VSO Writes	DASD Reads	DASD Writes	Castouts Scheduled
BANKC01	1	393	457	64	393	163

**** I/O Activities **** (SHARELVL 2/3)

Area Name	Share Level	Look-aside	CF Reads	CF Writes	Read Hits	Valid Reads	DASD Reads	DASD Writes	Castouts Scheduled
BANKC02	2	NO	393	457	-	-	64	393	163

**** DEDB Write IWAIT ****

ADSname	Share Level	VSO	IWAITs	Elap/IWAIT Sc.Mil.Mic	Max value Sc.Mil.Mic	Pct Tot IWAITs	Pct Tot IWT Elp	---- CI Write Count ---- CI/IWAIT	Max value
BANKC03	1	YES	426	4.404	140.204	49.53%	23.28%	2	8

**** Castout ****

Area Name	Shr Lvl	Start Time HH.MM.SS.TH	End Time HH.MM.SS.TH	Elapse Time Sc.Mil.Mic	CI Writes	Elapse Time Structure name 1 Sc.Mil.Mic	Elapse Time Structure name 2 Sc.Mil.Mic
BANKC04	2	15.03.09.85	15.03.09.90	52.019	150	BANKC00STR1 1.011	BANKC00STR2 1.045

**** CF I/O Wait ****

CF Structure Name	Found Count	Read Elap/IWAIT Sc.Mil.Mic	Max value Sc.Mil.Mic	Not Found	Write CI Count	Elap/IWAIT Sc.Mil.Mic	Max value Sc.Mil.Mic	Castout CI Writes	Elapsed/CI Sc.Mil.Mic	Max value Sc.Mil.Mic
BANKC00STR1	393	1.011	10.035	12	457	2.479	12.250	457	2.479	12.250

Summary

DEDBs perform well because of ..

- ❑ Shorter pathlengths
 - DB restrictions
 - Simplified buffer management
 - Simplified locking
- ❑ Asynchronous chained writes after commit (output threads)

But need to pay more attention to ..

- ❑ Database space utilization
- ❑ Buffer allocation and utilization
- ❑ Locking granularity
- ❑ Logging (LGNR)