



IBM Software Group

# Virtual Storage Constraint Relief in DB2 for z/OS V8 ... What to Expect

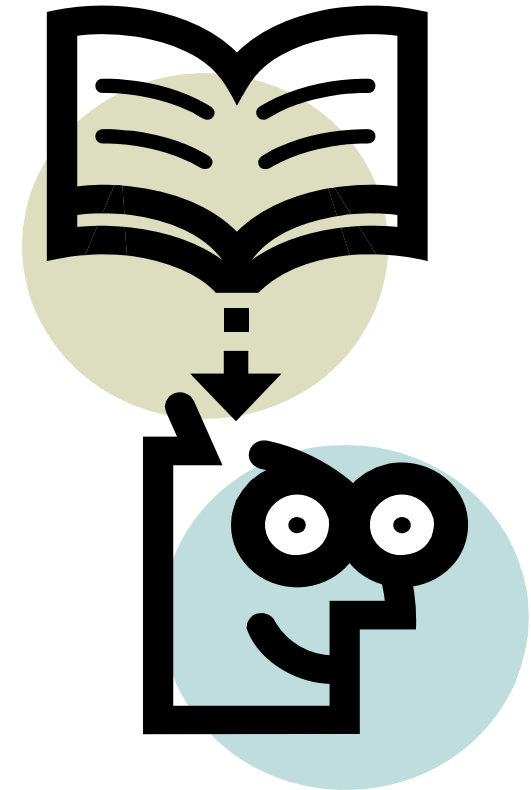
IBM Information Management software

John J. Campbell  
Distinguished Engineer  
DB2 for z/OS Development  
Email: [CampbelJ@uk.ibm.com](mailto:CampbelJ@uk.ibm.com)

 ON DEMAND BUSINESS™

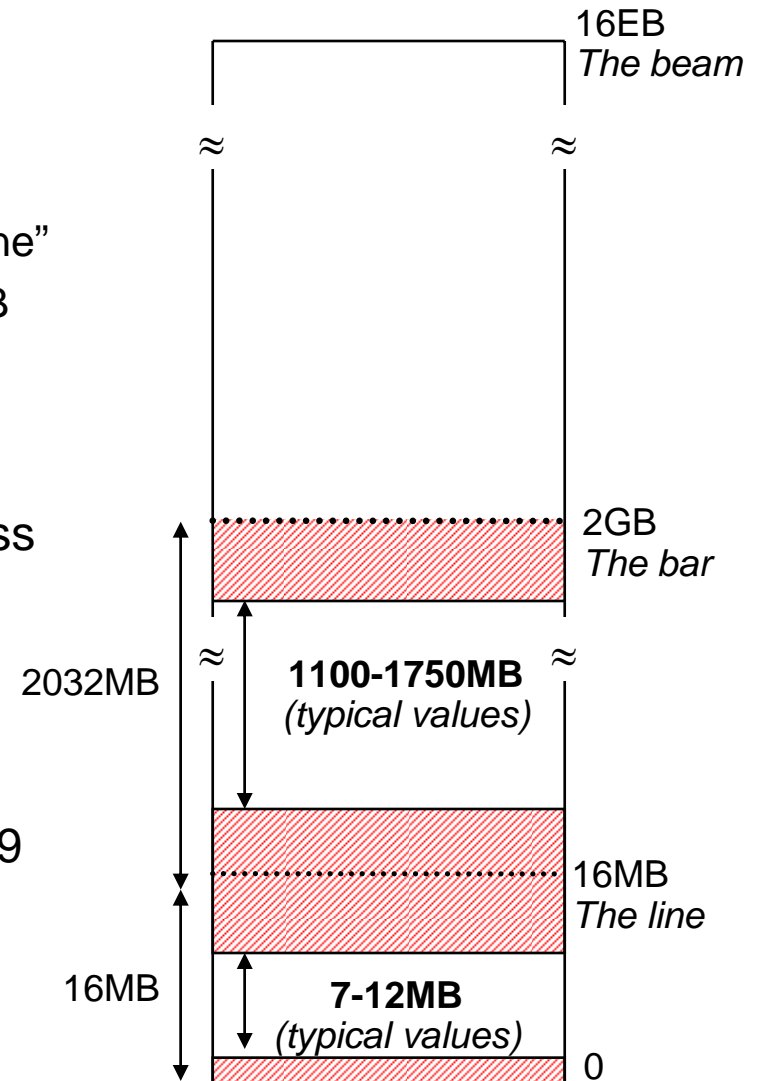
## Topics

- DBM1 64-bit Virtual Memory Map
- DBM1 64-bit Virtual and Thread Storage
- Projecting V8 Use from V7 Statistics Trace
- Key Messages
- Problem Recap and Driving Factors
- Analysing Virtual Storage Used
- Tuning Options
- Protecting The System
- Real Storage Use
- Summary

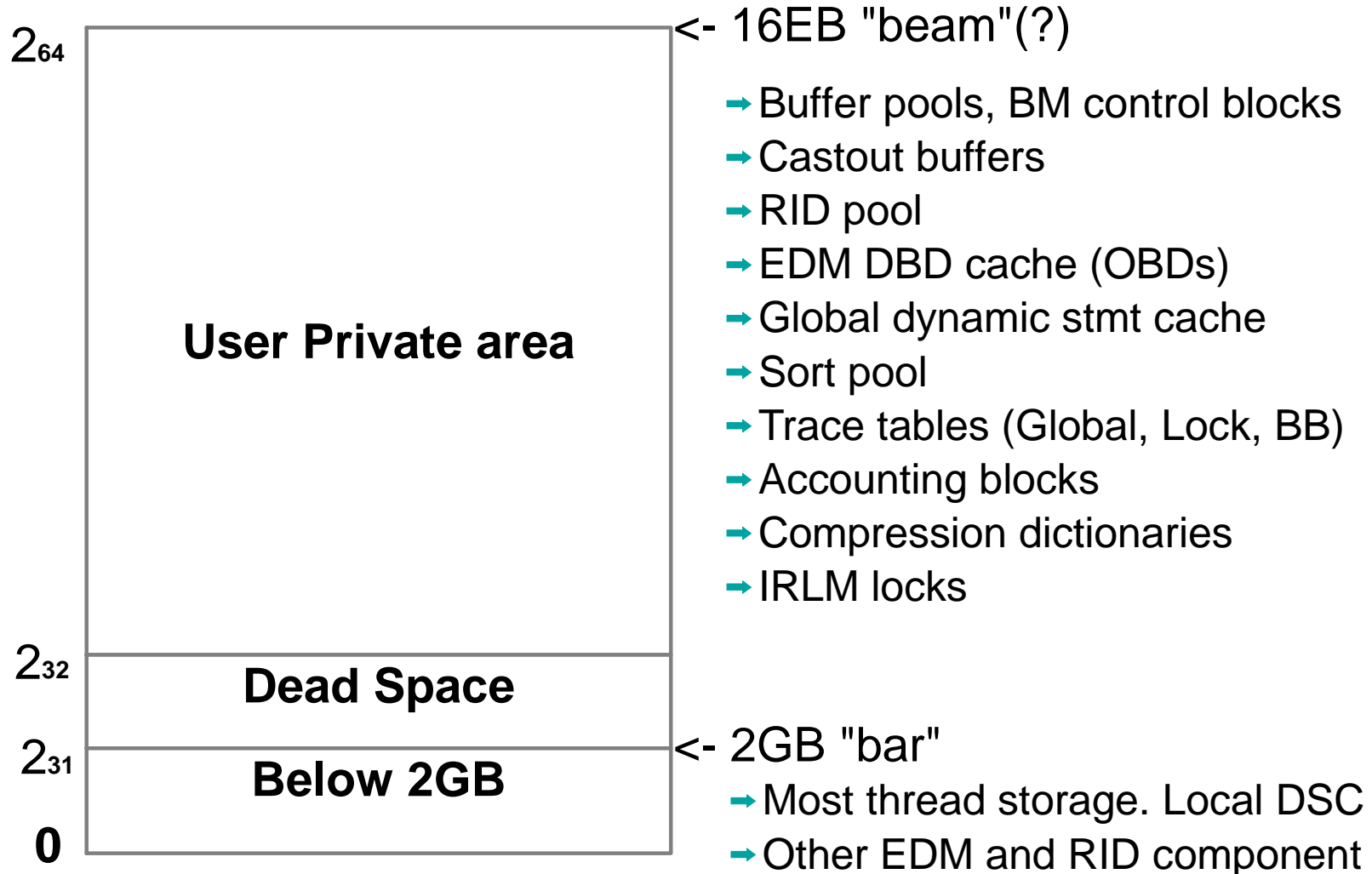


## DBM1 64 bit Virtual Memory Map

- Each address space now has an addressing range of 16EB (“the beam”) based on 64 bit addressing but
  - ▶ Maximum of 16MB available “below the 16MB line”
  - ▶ Maximum of 2032MB available “above the 16MB line” and “below the 2GB bar”
- Practical maximum available to DB2 and specifically DBM1 Address Space is much less
  - ▶ Typical 7-12MB available “below the line”
  - ▶ Typical 1100-1750MB available “above the line”
- Storage is allocated into different subpools which have unique characteristics e.g., SP229
  - ▶ Storage acquired via GETMAIN
  - ▶ Storage released by FREEMAIN



# DBM1 64 bit Virtual Memory Map ...



## DBM1 64-bit Virtual and Thread Storage

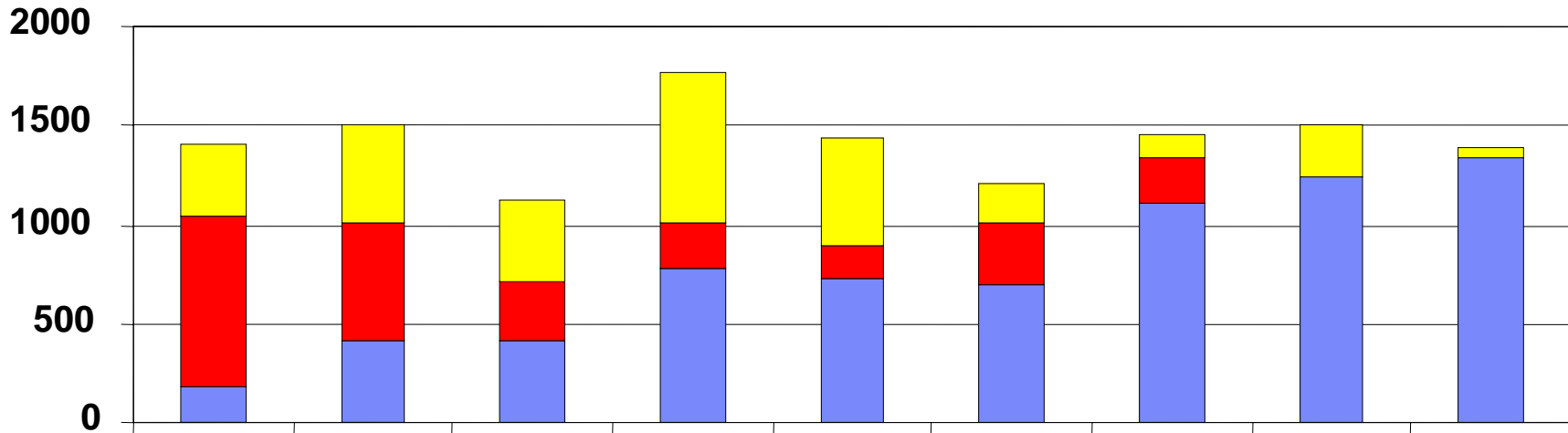
- Most of the thread storage stayed below the 2GB with regression
  - ▶ Agent Local
  - ▶ Getmained Stack Storage
  - ▶ Local Dynamic Statement Cache
- Regression estimates
  - ▶ Agent Local Storage:
    - System: +40% for system threads
    - Non-system: +30 to 40% for static, +50 to 100% for dynamic
  - ▶ Getmained Stack storage: +100%
  - ▶ Local Dynamic Stmt Cache Cntl Blks: -75%
  - ▶ Thread Copies of Cached SQL Stmts: +100%
  - ▶ EDM pool: +30 to 50% (\*)
  - ▶ RID pool: -75%
  - ▶ Others: -100%

# Projecting V8 Use from V7 Statistics Trace

	V7 measured (MB)	V8 estimated (MB)	Notes:
Virtual buffer pool	15	0	0
Buffer pool control blocks	95	0	0
Dataspace lookaside buffe	48	0	0
EDM pool	88	124	+30 to 50%
Compression dictionary	54	0	0
Castout buffers	28	0	0
System thread storage	89	125	+40%
User thread storage	114	217	+30 to 40% static SQL, +50 to 100% dynamic SQL
RDS OP pool	8	0	0
RID pool	78	20	-75%
Pipe Manager subpool	1	1	Same
Local Dynamic Control Blk	70	18	-75%
Local Dynamic Stmts	8	16	+100%
BM/DM trace table	18	9	-50%
Fixed storage	3	3	Same
Stack storage	58	116	+100%
<b>Total</b>	<b>775</b>	<b>649</b>	
% VSCR		16	

## Potential Value of V8 64-bit Virtual below 2GB bar

MB



Top of **Yellow** = Extended Region Size (Max)

Top of **Red** = V7 use

Top of **Blue** = V8 use

## Key Messages

- DBM1 64-bit will provide valuable VSCR for most installations
- Actual value of DBM1 will vary by installation
- But DBM1 64-bit support does not eliminate the problem ...
- Installations must continue to capacity plan for, monitor, tune and optimise use of virtual storage below 2GB bar
- Must have sufficient real storage to fully back increased total virtual storage usage: below 2GB bar and above the 2GB bar
- May be able to support some additional active threads?
- May be able to set zparms CONTSTOR=NO and MINSTOR=NO



## What Is The DBM1 Problem?

### GETMAIN processing by DB2

- ▶ Requests may be conditional or unconditional
- ▶ "Short on Storage" condition can occur for both
- ▶ DB2 recovery routines may be able to clean up
- ▶ Individual DB2 threads (allied, DBAT) may abend with 04E/RC=00E200xx when insufficient storage available
  - e.g., 00E20003 & 00E20016
- ▶ Eventually DB2 subsystem may abend with abend S878 or S80A when critical task and no toleration of error

## What Are The Drivers?

- Workload growth, both organic and through mergers & acquisitions
- Shrinking maximum region size (EPVT) available to DB2
  - ▶ Conflict between using ECSA (IMS) and EPVT (DB2)
  - ▶ Extensive use of ECSA by IMS across dependent regions
    - Mostly buffer pools, control blocks, data are in ECSA
    - Sizes are at user choice
    - For best performance they tend to be large
    - Not exploiting VSCR features of recent IMS releases
- LPAR consolidation
- Other use of extended common areas e.g., WebSphere
- Generous over allocation for safety of ECSA and other extended common areas
- Common LPAR image for Sysplex (best practice)

## What Are The Drivers? ...

- Increase in average thread footprint across successive DB2 releases
- Long running persistent threads (IMS WFI, CICS Protected Entry Threads, WebSphere Connection Pool, DDF Connection Pool)
- Plans/packages with RELEASE(DEALLOCATE)
- Use of Local (thread) Dynamic Statement Caching (BIND option KEEPDYNAMIC YES and zparm MAXKEEPD > 0) to reduce CPU consumption
  - ▶ WebSphere with JDBC to avoid short prepares
  - ▶ ERP & CRM applications e.g., SAP
- CTHREAD and MAXDBAT throttles set to high values that cannot be supported when system slowdown occurs, workload keeps arriving and more threads come into play
- Degradation in DB2 Activity Time (Acctg Class 2 Elapsed)
  - ▶ Degraded IO response and CF service times
  - ▶ Application lockouts

## Deadly Combination

- Maximum use of Dataspace Bufferpool under V7
- Many concurrent persistent threads
- KEEP DYNAMIC(YES)
- RELEASE(DEALLOCATE)
- CTHREAD and/or MAXDBAT throttles wide open
- V6->(V7)->V8

## Thread Footprint

- Low end 200 to 400KB
  - ▶ Simple Static SQL
- Mid range 500 KB to 2MB
  - ▶ Most Static / Simple Dynamic SQL
- High end 3MB to 10MB plus
  - ▶ Complex Dynamic SQL, Heavy Sort, Parallelism
- Mileage will vary by installation

## Analysing Virtual Storage Used

- RMF for very high level view
  - ▶ Virtual Storage (VSTOR) Private Area Report
    - Interval data collected in SMF Type 78-2
    - Collected by RMF Monitor I session option: **VSTOR(D,xxxxDBM1)**
    - Produced by RMF Post Processor option: **REPORTS(VSTOR(D,xxxxDBM1))**
  - ▶ Use to identify potential storage shortages and to get historical view of virtual storage consumption
  - ▶ Calculate amount of storage available above the line by subtracting **MAX LSQA/SWA/229/230 PAGES ALLOCATED** and **MAX USER REGION PAGES ALLOCATED** from **REGION ASSIGNED**
  - ▶ How much is enough?
    - Greater than 500MB spare is AOK (**GREEN**)
    - Between 200-500MB spare is boundary condition (**AMBER**)
    - Less than 200MB action is required (**RED**)

V I R T U A L   S T O R A G E   A C T I V I T Y

z/OS V1R5                      SYSTEM ID X9                      DATE 05/25/2005                      INTERVAL 10.00.68  
 RPT VERSION V1R5 RMF                      TIME 15.00.00                      CYCLE 0.250 SECONDS

PRIVATE AREA SUMMARY

JOB NAME -            DBL1DBM1                                      REGION REQUESTED                      0K  
 STEP NAME -        DBL1DBM1                                      REGION ASSIGNED (BELOW 16M)    11.0M  
 PROGRAM NAME -    DSNYASCP                                      **REGION ASSIGNED (ABOVE 16M) 1635M**  
 NUMBER OF SAMPLES - 240

PRIVATE STORAGE MAP

BELOW 16M			EXTENDED (ABOVE 16M)		
AFFFFF	LSQA/SWA			LSQA/SWA	7FFFFFFF
	229/230	328K	BOTTOM OF	229/230	1089M
AAE000	15.00.00		ALLOCATED AREA	15.05.32	3BEF700
	UNUSED	0K		UNUSED	0K
B00000			GETMAIN LIMIT		7FFFFFFF
	UNUSED	10.4M		UNUSED	521M
3F000	15.00.00		TOP OF	15.00.00	1B58F00
	USER		ALLOCATED AREA	USER	
	REGION	228K		REGION	
6000					24.6M
	SYSTEM REGION	16K			
2000					19D0000

	BELOW 16M			ABOVE 16M		
	MIN	MAX	AVG	MIN	MAX	AVG
<b>LSQA/SWA/229/230</b>						
FREE PAGES (BYTES)	40K 15.00.00	40K 15.00.00	40K	28K 15.02.37	724K 15.00.00	210K
LARGEST FREE BLOCK	36K 15.00.00	36K 15.00.00	36K	4K 15.02.37	320K 15.00.00	86K
<b>PAGES ALLOCATED</b>						
(IN BYTES)	288K 15.00.00	288K 15.00.00	288K	92.8M 15.00.00	<b>1089M 15.05.32</b>	828M
<b>USER REGION</b>						
FREE PAGES (BYTES)	10.4M 15.00.00	10.4M 15.00.00	10.4M	521M 15.05.32	1517M 15.00.00	783M
LARGEST FREE BLOCK						
IN GETMAIN LIMIT	10.4M 15.00.00	10.4M 15.00.00	10.4M	521M 15.05.32	1517M 15.00.00	783M
<b>PAGES ALLOCATED</b>						
(IN BYTES)	228K 15.00.00	228K 15.00.00	228K	24.6M 15.00.00	<b>24.6M 15.00.00</b>	24.6M

**1635 - (1089 + 24.6) = 521.4MB available**

## Analysing Virtual Storage Used ...

What consumes the virtual storage used by DBM1 address space?

- DB2 instrumentation for detail
  - ▶ IFCID 225
    - Summary Information
    - Snapshot as each DB2 Statistics interval becomes due
    - Available through DB2 Statistics Trace Class 1
    - Tiny overhead in terms of increased CPU resource consumption and increased SMF data volume
    - Start automatically via zparm SMFSTAT(1,...)
    - Recommend zparms STATIME=5 and SYNCVAL=0
  - ▶ IFCID 217
    - Detail Information at thread level
    - Available through Global Trace Class 10
  - ▶ For description of IFCIDs see DSN810.SDSNIVPD(DSNWMSGGS)



## Analysing Virtual Storage Used ...

What consumes the virtual storage used by DBM1 address space? ...

- First class support provided by OMEGAMON XE for DB2 PM/PE, DB2 PM and DB2 PE
  - ▶ Statistics Trace | Report
    - Includes FILE and LOAD data base table support as well as upgrade (ALTER TABLE ....) of already installed table DB2PM\_STAT\_GENERAL
  - ▶ Record Trace Report
    - IFCID 217 and 225 supported independent of DB2 release which created records
  - ▶ New SPREADSHEETDD subcommand option
    - Introduced with APAR PK31073
    - Available now for both DB2PE V2.1 and DB2PM V8.1
- REXX Tools (MEMU2, MEMUSAGE) with User Guide
  - ▶ Available for download from DB2 Trading Post off DB2 for z/OS Home Page

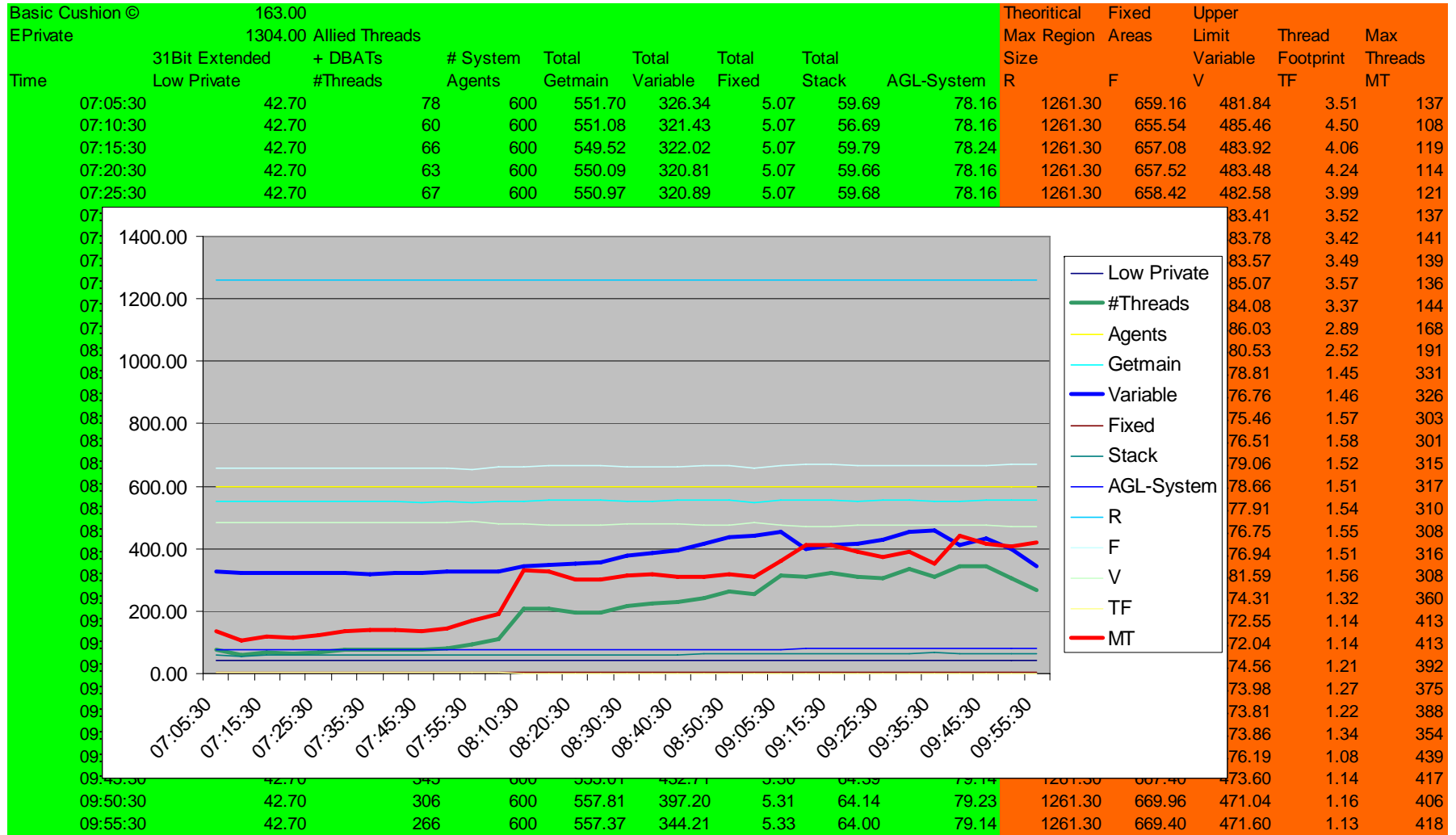
# Sample – Statistics Trace | Report

DBM1 AND MVS STORAGE BELOW 2 GB		QUANTITY	DBM1 AND MVS STORAGE BELOW 2 GB		QUANTITY
<b>TOTAL DBM1 STORAGE BELOW 2 GB</b>	<b>(MB)</b>	<b>773.05</b>	24 BIT LOW PRIVATE	(MB)	0.23
<b>TOTAL GETMAINED STORAGE</b>	<b>(MB)</b>	<b>575.00</b>	24 BIT HIGH PRIVATE	(MB)	2.25
VIRTUAL BUFFER POOLS	(MB)	429.69	<b>31 BIT EXTENDED LOW PRIVATE</b>	<b>(MB)</b>	<b>27.38</b>
VIRTUAL POOL CONTROL BLOCKS	(MB)	13.43	31 BIT EXTENDED HIGH PRIVATE	(MB)	954.23
EDM POOL	(MB)	117.19	<b>EXTENDED REGION SIZE (MAX)</b>	<b>(MB)</b>	<b>1714.00</b>
COMPRESSION DICTIONARY	(MB)	2.35	<b>EXTENDED CSA SIZE</b>	<b>(MB)</b>	<b>200.06</b>
CASTOUT BUFFERS	(MB)	9.13			
DATA SPACE LOOKASIDE BUFFER	(MB)	0.00	<b>AVERAGE THREAD FOOTPRINT</b>	<b>(MB)</b>	<b>3.61</b>
HIPERPOOL CONTROL BLOCKS	(MB)	0.05	<b>MAX NUMBER OF POSSIBLE THREADS</b>		<b>236.12</b>
DATA SPACE BP CONTROL BLOCKS	(MB)	0.00			
<b>TOTAL VARIABLE STORAGE</b>	<b>(MB)</b>	<b>139.53</b>			
<b>TOTAL AGENT LOCAL STORAGE</b>	<b>(MB)</b>	<b>53.94</b>			
<b>TOTAL AGENT SYSTEM STORAGE</b>	<b>(MB)</b>	<b>32.35</b>			
NUMBER OF PREFETCH ENGINES		77.00			
NUMBER OF DEFERRED WRITE ENGINES		300.00			
NUMBER OF CASTOUT ENGINES		73.00			
NUMBER OF GBP WRITE ENGINES		58.00			
NUMBER OF P-LOCK/NOTIFY EXIT ENGINES		9.00			
<b>TOTAL AGENT NON-SYSTEM STORAGE</b>	<b>(MB)</b>	<b>21.60</b>			
<b>TOTAL NUMBER OF ACTIVE USER THREADS</b>		<b>29.67</b>			
RDS OP POOL	(MB)	34.54			
RID POOL	(MB)	16.97			
PIPE MANAGER SUB POOL	(MB)	0.00			
LOCAL DYNAMIC STMT CACHE CNTL BLKS	(MB)	0.99			
<b>THREAD COPIES OF CACHED SQL STMTS</b>	<b>(MB)</b>	<b>0.00</b>			
IN USE STORAGE	(MB)	N/A			
STATEMENTS COUNT		N/A			
HWM FOR ALLOCATED STATEMENTS	(MB)	N/A			
STATEMENT COUNT AT HWM		N/A			
DATE AT HWM		N/A			
TIME AT HWM		N/A			
BUFFER & DATA MANAGER TRACE TBL	(MB)	9.41			
<b>TOTAL FIXED STORAGE</b>	<b>(MB)</b>	<b>3.80</b>			
<b>TOTAL GETMAINED STACK STORAGE</b>	<b>(MB)</b>	<b>54.71</b>			
STORAGE CUSHION	(MB)	112.04			

# DBM1 Storage from Statistics Trace | Report

TOTAL DBM1 STORAGE USED	TOTAL GETMAINED STORAGE	VIRTUAL BUFFER POOLS EDM POOL COMPRESSION DICTIONARY CASTOUT BUFFERS DATASPACE LOOKASIDE BUFFER
	TOTAL VARIABLE STORAGE	TOTAL AGENT SYSTEM STORAGE TOTAL AGENT LOCAL STORAGE RDS OP POOL RID POOL PIPE MANAGER SUB POOL LOCAL DYNAMIC STMT CACHE CTL BLKS LOCAL DYNAMIC STMT CACHE STMT POOL BUFFER & DATA MANAGER TRACE TBL VIRTUAL POOL CONTROL BLOCKS HIPERPOOL CONTROL BLOCKS DATASPACE BP CONTROL BLOCKS
	TOTAL FIXED STORAGE	
	TOTAL GETMAINED STACK STORAGE	
	STORAGE CUSHION	

# Study Historical Evolutionary Trend



## Tuning Options

- Full System Contraction (“Reserve Parachute”)
- Turn on Thread Storage Contraction (CONTSTOR=YES)
- Turn on “Best Fit” algorithm for Thread Storage (MINSTOR=YES)
- Reduce size of Local Dynamic Statement Cache (MAXKEEPD)
- Invalidate statements from the Local Statement Cache
- Super Size Bufferpools and More ESA Compression
- Reduce use of RELEASE(DEALLOCATE)
- Reduce size of Extended Common Areas (ECSA, EPLPA, etc)
- Reduce number of long running persistent threads
- Exploit Type 2 Inactive Connections for DDF work
- Switch SMF INTERVAL recording for STCs to NODETAIL
- Implement Data Sharing, increase width of existing Data Sharing Group, or distribute workload around existing Data Sharing Group

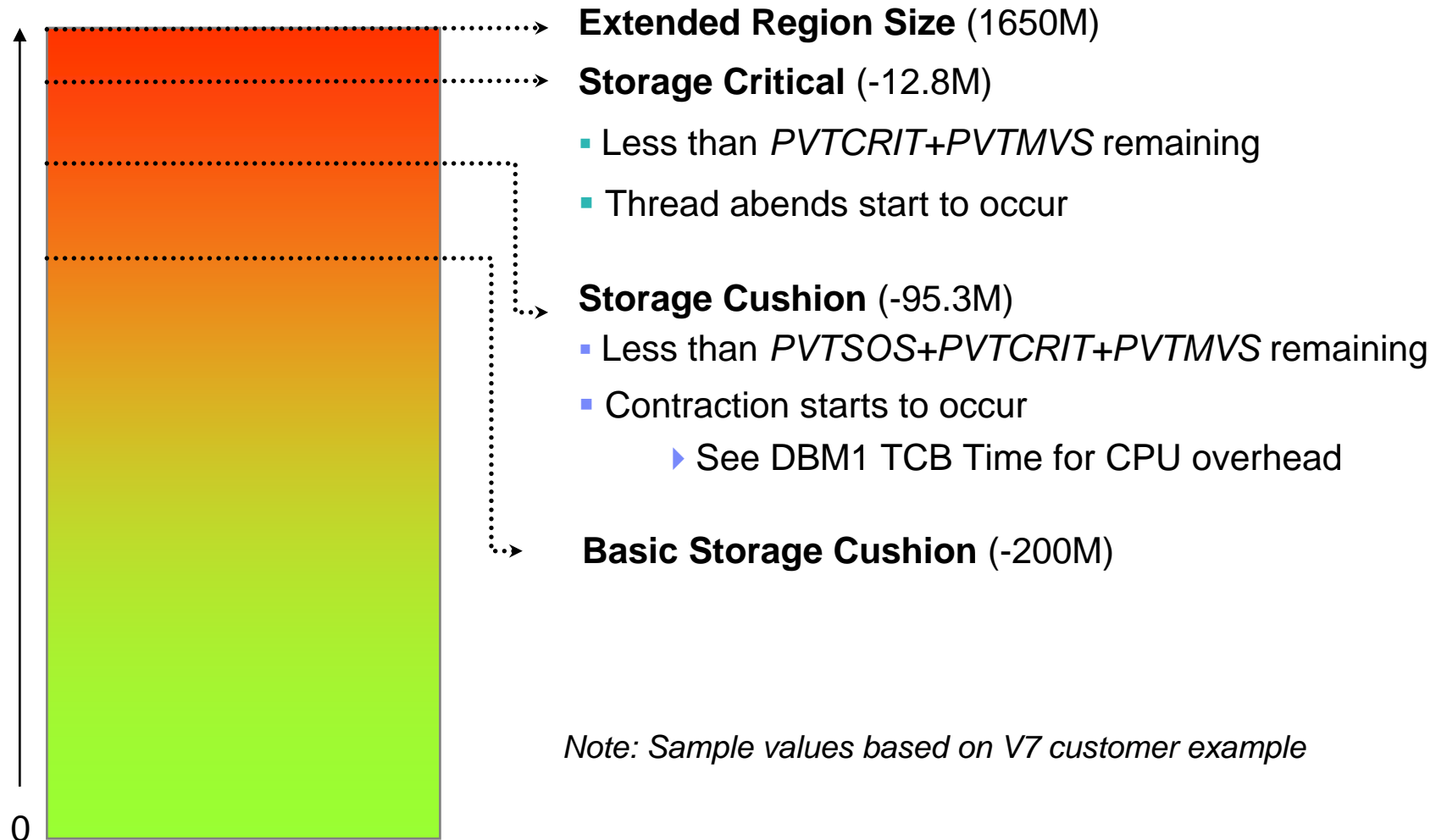
# Full System Storage Contraction

Driven by

- ▶ **PVTCRIT** = MVS Cushion = Storage for Must Complete
  - Fixed, real value
  - Based on CTHREAD/MAXDBAT
  - $(CTHREAD+MAXDBAT+1) * \underline{64K}$
- ▶ **PVTMVS** = MVS Available = Storage for MVS
  - Amount set aside (reserved) for dataset opens
  - Based on DSMAX
  - “Reserve” – decrease on open and increase on close
  - $(DSMAX*1300)+40K$
- ▶ **PVTSOS** = MVS Warning To Contract = Cushion Warning
  - Max(5% of Extended Region Size, PVTCRIT)

**Do not oversize CTHREAD and MAXDBAT as it inflates Storage Cushion ( $PVTSOS+PVTCRIT+PVTMVS$ )**

# Full System Storage Contraction ...



*Note: Sample values based on V7 customer example*

## Thread Storage Contraction

- Turned on by zparm CONTSTOR = YES
- Associated CPU overhead (typical < 1-2%)
- Design point is long running persistent threads with RELEASE(COMMIT)
- Compresses out part of Agent Local Non-System storage
- Does not compress
  - ▶ Agent Local System
  - ▶ Getmained Stack Storage
  - ▶ Local Dynamic Statement Cache
- Controlled by two hidden zparms
  - ▶ SPRMSTH @ 1048576 (1MB)
  - ▶ SPRMCTH @ 10 (commits)
- Triggered at:
  - ▶ # Commits > SPRMCTH | (Agent Local Non-System > SPRMSTH & # Commits > 5)



## Best Fit Algorithm for Thread Storage

- With zparm MINSTOR=NO (default), first fit algorithm is used
  - ▶ Fragmentation may happen with free space (leap frog effect)
- With zparm MINSTOR=YES, best fit algorithm is used instead
  - ▶ Will go through all the chains across all the segments
  - ▶ Makes the storage denser
  - ▶ Observed CPU overhead < 1% for 3-4MB storage pools
  - ▶ Danger is that it masks storage leaks (makes them appear to go away)
  - ▶ It makes debugging storage leaks more difficult
  - ▶ Also degraded slower performance as the storage leak progresses
- Use zparm MINSTOR=YES when
  - ▶ System is fully tuned and optimised for storage
  - ▶ You have determined there are no leaks
  - ▶ Out of other options and need the last ounce of storage

## Reduce size of Local Dynamic Statement Cache

- Goal is to reduce storage requirement below the 2GB bar for Thread Copies of Cached SQL Stmts when using KEEP DYNAMIC YES
- But this increase in number “short prepares” and associated increase in CPU resource consumption
- Increase size of EDM Prepared Statement Cache above the 2GB bar to compensate by trying to reduce the number of “full prepares” and offset the increase in CPU resource consumption
- Reduce zparm MAXKEEPD incrementally, and
- Increase size of EDM Prepared Statement Cache above the 2GB bar

# Invalidate statements from the Local Statement Cache

- Least Recently Prepared Statements are thrown away from the cache at commit based on MAXKEEPD with KEEP DYNAMIC YES
- APAR PK21861 introduced new zparm CACHEDYN\_FREELOCAL
- Ahead of commit based on internal thresholds (subject to change) will invalidate statements from the Local Statement Cache and release the associated storage
- Statements will be purged at end of section
- CACHEDYN\_FREELOCAL settings

0 = off (default)

1 = If (LDSC  $\geq$  500MB & DBM1 Used  $\geq$  75%) then free  $\geq$  100KB statement | If DBM1 Used  $\geq$  85% then free any statement

2 = If (LDSC  $\geq$  500MB & DBM1 Used  $\geq$  80%) then free  $\geq$  100KB statement | If DBM1 Used  $\geq$  88% then free any statement

3 = If (LDSC  $\geq$  350MB & DBM1 Used  $\geq$  75%) then free  $\geq$  100KB statement | If DBM1 Used  $\geq$  88% then free any statement

## Reduce use of RELEASE(DEALLOCATE)

- Use RELEASE DEALLOCATE selectively based on benefit
  - ▶ Overuse with persistent threads can create a virtual storage issue
    - Accumulating ever more storage for statements that are not being re-used
    - Storage for unused statements can be left around until deallocation
    - Also drive up demand for EDM Pool resources
    - Ineffective thread and full system storage contraction
  - ▶ Best reserved for
    - High volume and/or performance sensitive at reasonable volume OLTP plans/packages
    - Long running batch programs that take frequent intermediate commits

## Supersize Bufferpools and ESA Compression

- Super size bufferpools provided fully backed with large real storage
  - ▶ Potential for better bufferpool hit ratio
    - Reduce # sync IO waits
    - Reduce DB2 Activity Time
    - Fewer threads to maintain same throughput (VSCR)
- More use of ESA Compression provided fully backed by real storage
  - ▶ Less DASD space
  - ▶ Faster sequential scan
  - ▶ Potential for better bufferpool hit ratio
    - Reduce # sync IO waits
    - Reduce DB2 Activity Time
    - Fewer threads to maintain same throughput (VSCR)

## Other Tuning Options

- Reduce appetite for Extended Common Areas (ECSA, ...) and give back to get larger Extended Region Size (Max) for DB2
  - ▶ IMS users should exploit VSCR features of IMS V7 and later releases
  - ▶ Avoid excessive over allocation
- Reduce number of long running persistent DB2 threads
  - ▶ If over configured to meet throughput requirement
    - Reduce the number of such threads
  - ▶ Ruthlessly cut back on number of JDBC/SQLJ Data Sources
    - Collapse out redundant Data Sources
- Exploit Type 2 Inactive Connections for DDF work
  - ▶ Do not use KEEP DYNAMIC(YES) !!!
  - ▶ Close open held cursors ahead of commit
  - ▶ etc

## Other Tuning Options ...

- Switch SMF INTERVAL recording for STCs to NODETAIL
  - ▶ Problem
    - See Information APAR II07124
    - Symptom SP230 Key storage increasing x MB per day
    - Caused by the amount of SMF Record Type 30 Subtype 4 and Subtype 5 data filling SP230
  - ▶ Solution
    - Change from DETAIL to NODETAIL for STC in SMFPRMxx
    - May have to rewrite accounting programs if they are using Subtype 4 and Subtype 5
- Implement Data Sharing, increase width of existing Data Sharing Group, or redistribute work over existing members of Data Sharing Group
  - ▶ Redistribute and spread user workload over multiple members
  - ▶ Fewer active threads (allied, DBATs) per member

## Preventative Maintenance

- Monitor DB2 Storage Information APAR on a weekly basis and apply as preventative service
  - ▶ See Info APAR II10817
- Some important storage related APARs to highlight
  - ▶ PK21237
    - Reset castout and notify exit engines. Also reduce the number of deferred write engines, GBP write engines, and castout engines.
  - ▶ PK21268
    - Move Current Path storage out of stack and above 2GB
  - ▶ PK21892
    - Reduce stack storage for ND type CICS threads
  - ▶ PK22442
    - DDF address space storage shortage while processing distributed threads with hundreds of output columns causing large SQLDA
  - ▶ PK21861
    - Local dynamic statement cache cleanup when infrequent commit and/or high concurrent full Prepare



## Protecting The System

- Plan to keep 200MB spare
  - ▶ Avoid hitting short on storage and driving Full System Storage Contraction
  - ▶ Provide some headroom for:
    - Tuning, some growth, Fast Log Apply, abnormal operating conditions
    - Estimate Maximum Number of Threads that can be supported
      - Allied
      - DBAT
- Set zparms CTHREAD and MAXDBAT to protect the system
  - ▶ Theoretical maximum:  $CTHREAD + MAXDBAT = 2000$
  - ▶ Practical maximum is much less (typical range 300-850)
  - ▶ Avoid over committing resources
  - ▶ Deny service and queue work outside the system to keep system alive

# Estimating Maximum Number of Threads

“Basic” Formula for estimating Number of Active Threads

Working Max = Extended Region Size  
minus 31bit Extended Low Private  
minus 200MB (Basic Cushion)

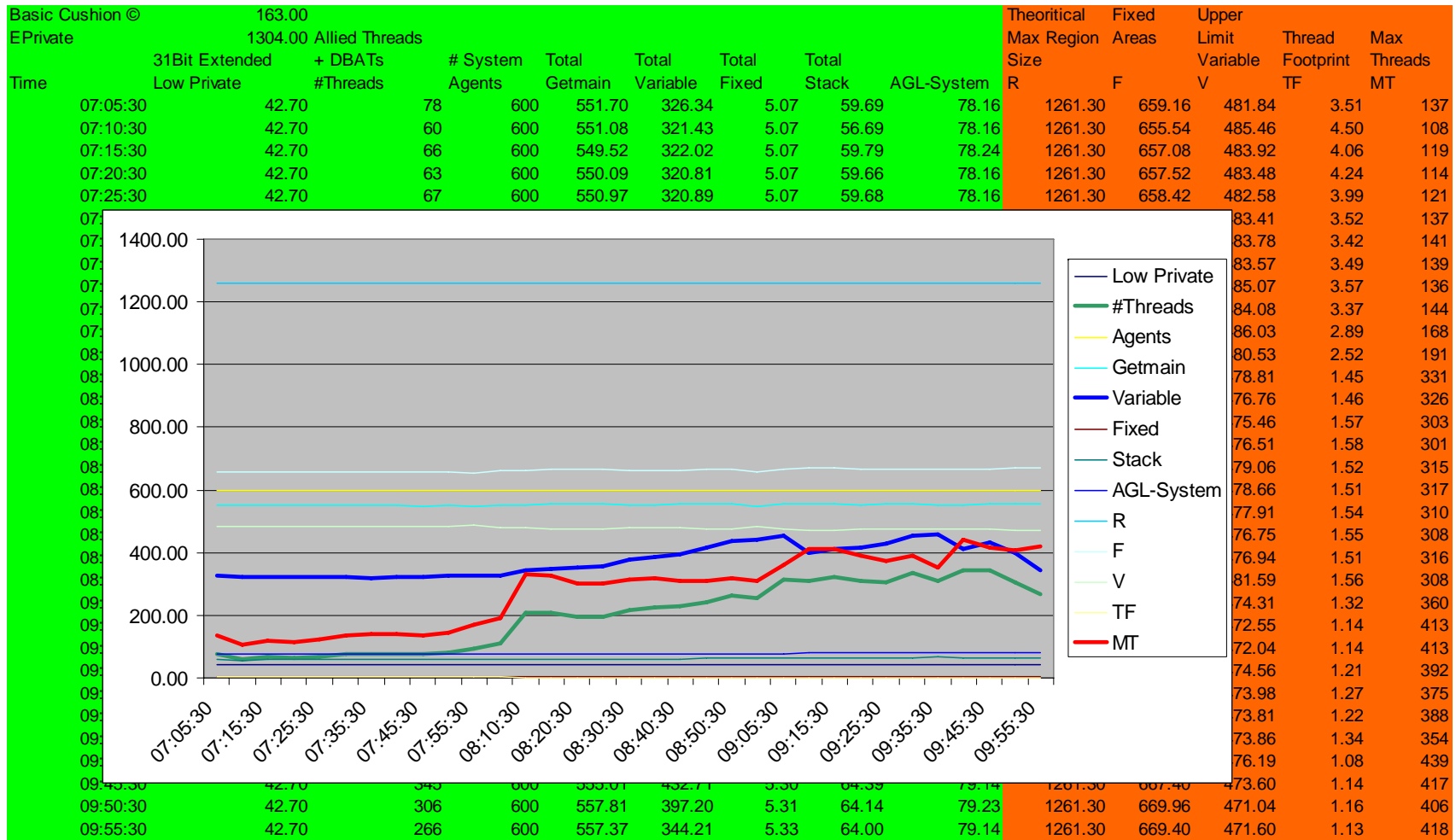
Fixed Areas = Total Getmained Storage below the 2GB bar  
plus Total Getmained Stack Storage  
plus Total Fixed Storage

Upper Limit Variable = Working Max minus Fixed Areas

Thread Footprint = (Total Variable Storage minus Total Agent System Storage)  
/ (Allied Threads plus Current Active DBATs)

Max. No. of Active Threads = Upper Limit Variable / Thread Footprint

# Estimating Maximum Number of Threads ...



## Real Storage Use

- Important subsystems such as DB2 should not be paging IN from auxiliary storage (DASD)
  - ▶ Recommendation to keep page in rates low (near zero)
  - ▶ Monitor using RMF Mon III
- Backing rate is dense for 31-bit storage (as before in V7)
- Common misconception!
  - ▶ Backing rate is low for 64-bit storage (<10%)
- Increase in real storage with V8 is to be expected as control blocks bigger for 64-bit storage
  - ▶ Stack +100%
  - ▶ System Threads +40%
  - ▶ User Threads: +30-40% static, +50-100% dynamic
  - ▶ ...

plus above the 2GB bar storage management!
- Have observed significant growth in real storage demand in some installations

## Real Storage Use ...

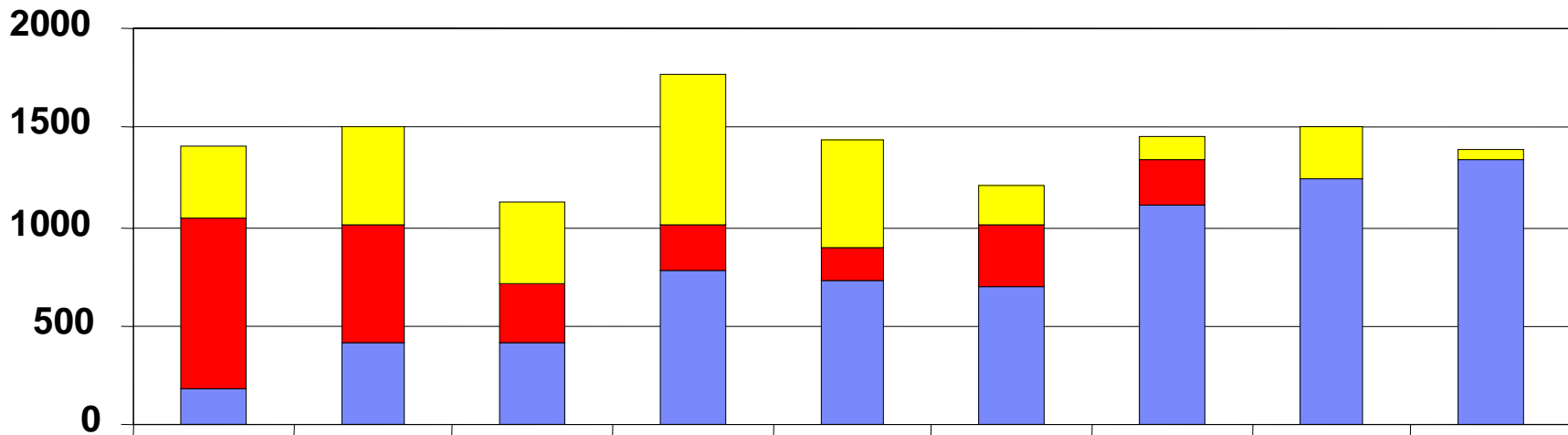
- Recommendation to apply critical must have service
  - ▶ PK19769
    - Add MVS discard to pool reset
  - ▶ OA15666 and PK25427
    - Discard real frames without hitting AVQLOW condition
  - ▶ PK21237
    - Drop number of Write Engines back to pre V8 levels
    - Single 64-bit pool for all Buffer Manager engines
  - ▶ PK21892
    - Throw away cached stack on thread deallocation
  - ▶ PK25326
    - Contract PLOCK engines after use

## Real Storage Use ...

- How to estimate real storage requirement for DBM1
  - ▶ Method assumes critical maintenance has already been applied
  - ▶ Subtract out fixed areas
    - Bufferpools, EDM Pools, ...
  - ▶ Below the 2GB bar
    - Assume  $V=R$  (Virtual = Real)
    - $V7 \rightarrow V8$ 
      - Stack +100%
      - User Threads +30-40% static, +50-100% dynamic
      - System Threads +40%
      - ...
  - ▶ Above the 2GB bar
    - Assume 1MB per active thread (pessimistic)
  - ▶ Add back the fixed areas

## Summary - Value of V8 64-bit Virtual below 2GB bar

MB



Top of **Yellow** = Extended Region Size (Max)

Top of **Red** = V7 use

Top of **Blue** = V8 use

## Summary ...

- DBM1 64-bit Virtual Memory Map
- DBM1 64 Virtual and Thread Storage
- Projecting V8 from V7 Statistics Trace
- Key Messages
- Problem Recap and Driving Factors
- Analysing Virtual Storage Used
- Tuning Options
- Protecting The System
- Real Storage Use
- Summary

