# From VIRTUALly Constrained to REALly Overcommitted

Adrian Burke
Db2 SWAT team SVL
agburke@us.ibm.com

# Agenda

- Storage terms and concepts
- Address spaces and DB2
- REAL and AUX impact on DB2
- Buffer Pools
- LFAREA
- Flash Express

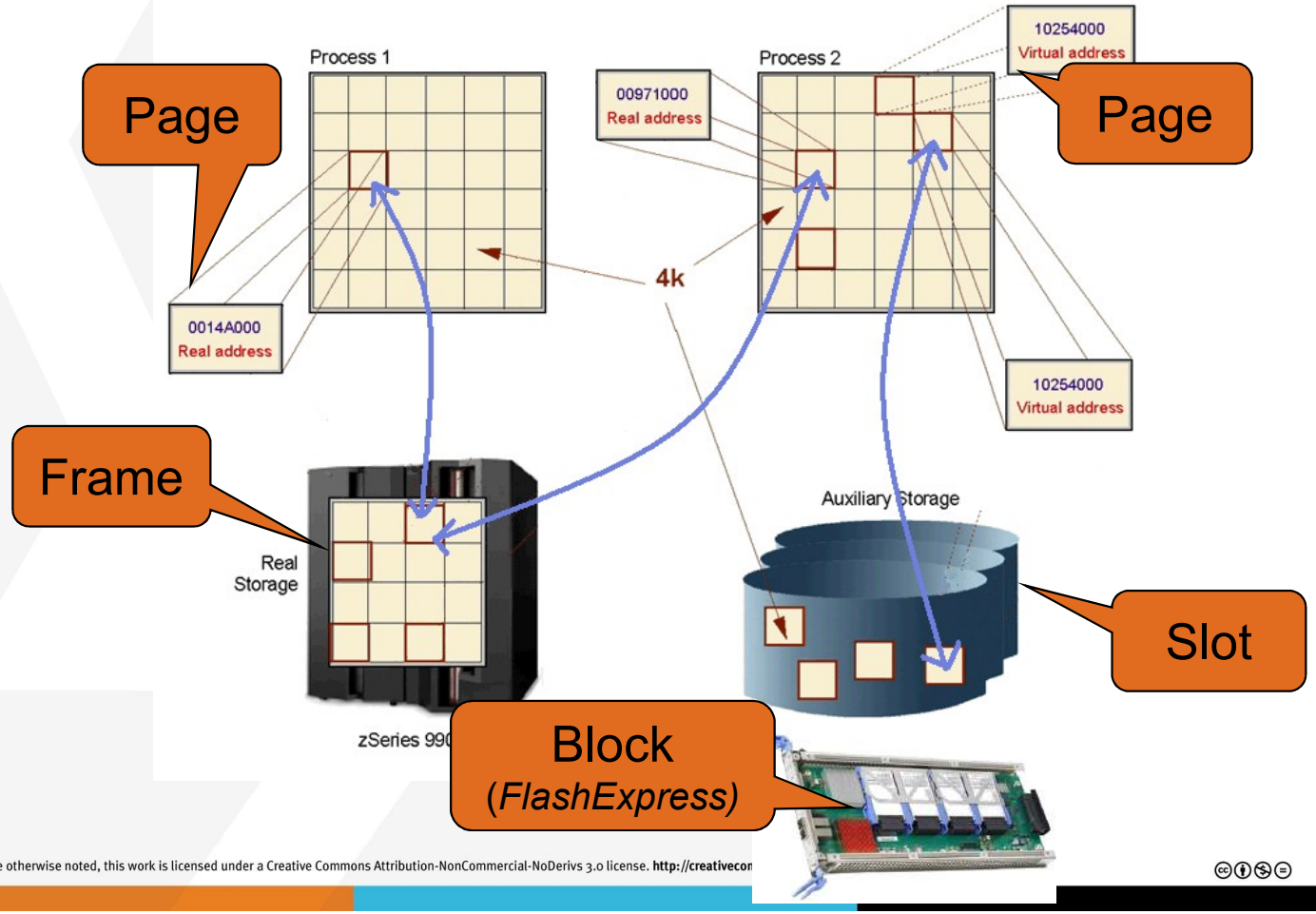
***Gartner named Db2 12 for z/OS an 'in-memory' database for the first time***

# How virtual storage works

- Virtual storage is divided into 2GB – 8GB *regions* composed of 1-megabyte *segments* composed of 4-kilobyte *pages*
  - z/OS uses region, segment and page tables to keep track of pages
- Addresses are translated dynamically, a process called *Dynamic Address Translation* (DAT)
- When a requested page is not in real storage, an interruption (called a *page fault*) is signaled and the page is brought into real
- Transfer of pages between auxiliary storage and real storage is called *paging*
- *Frames, slots, and blocks* are repositories for a page of information
  - Page is a 4k, 8k, 16k, 32k, 1MB, 2GB grouping representing frames (Virtual Storage)
  - A frame is a 4K, 1MB, 2GB piece of real storage (Central storage)
  - A slot is a 4K record in a page data set (AUX)
  - A block is a 4k record in FlashExpress (SCM)
    - Z14 has VFM (Virtual Flash Memory)

# How virtual storage works (continued…)

- We can only read/write to virtual page if it is in REAL memory, so we need to find it or page it in
  - You can have more than 1 virtual address that maps back to a 1 frame of real storage
  - Virtual address is not the same as real address



# Available Frame Queue Processing

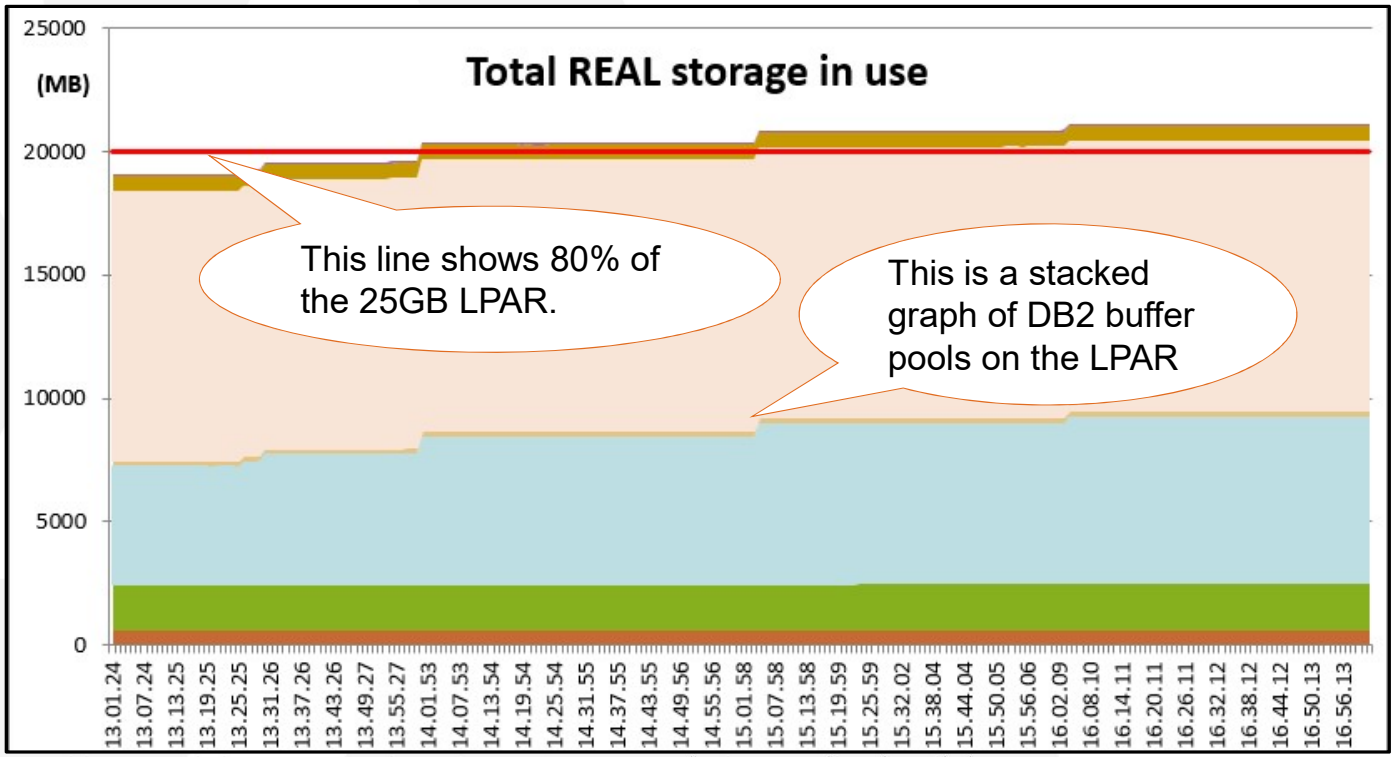
- When the supply of REAL frames becomes low, z/OS uses *page stealing* to replenish it based on an LRU mechanism
  - The unreferenced interval count (UIC) tracks how long it has been since the oldest frame was referenced (not effective for large LPARs)
- There is a 'low' level where stealing begins, and 'ok' value above which stealing stops
  - **MCCAFCTH=(*lowvalue*,*okvalue*) defaults in IEAOPTxx**
    - LOW will vary between MAX(MCCAFCTH lowvalue, 400, 0.2% of pageable storage)
    - OK will vary between MAX(MCCAFCTH okvalue, 800, 0.4% of the pageable storage)
      - RSM will automatically adjust the actual threshold values based on measurements of storage usage but doesn't let values get lower than MCCAFCTH low threshold
    - Typically no need to specify this parameter

# Messages to be aware of

- **REAL Storage – (too much fixed, not enough pageable)**
  - Informational level **50%**
    - Issue ENF 55
    - Issue IRA405I: % of real is fixed
  - Shortage level **80% if LPAR <320GB else 64GB**
    - Issue ENF 55
    - Issue IRA400E: shortage of pageable frames and list of top 20 consumers
    - Issue IRA403E swapping culprit out (in-real, moving it higher)
    - Issue IRA410E set non-swappable address space as non-dispatchable (STORAGENSWDP=YES)
  - Critical Shortage level (everything seen in Shortage +) **90%**
    - Issue IRA401E Critical paging shortage
    - If this case continues for 15 seconds issue IRA420I and IRA421D to show largest consumers and allow operator to cancel them

# AUTOSIZE Buffer Pools

- AUTOSIZE(YES) allows WLM to alter the BPs automatically +/- 25% each time DB2 is recycled, in order to avoid sync I/O delay
  - Customer had this on, and PGFIX(YES) for all buffer pools
  - z/OS 2.1 allows them to shrink
  - V11 allows for a MINSIZE and MAXSIZE to be set
- Over time the BPs grew to consume 80% of the LPAR causing address spaces to be non-dispatchable = system slowdown!!

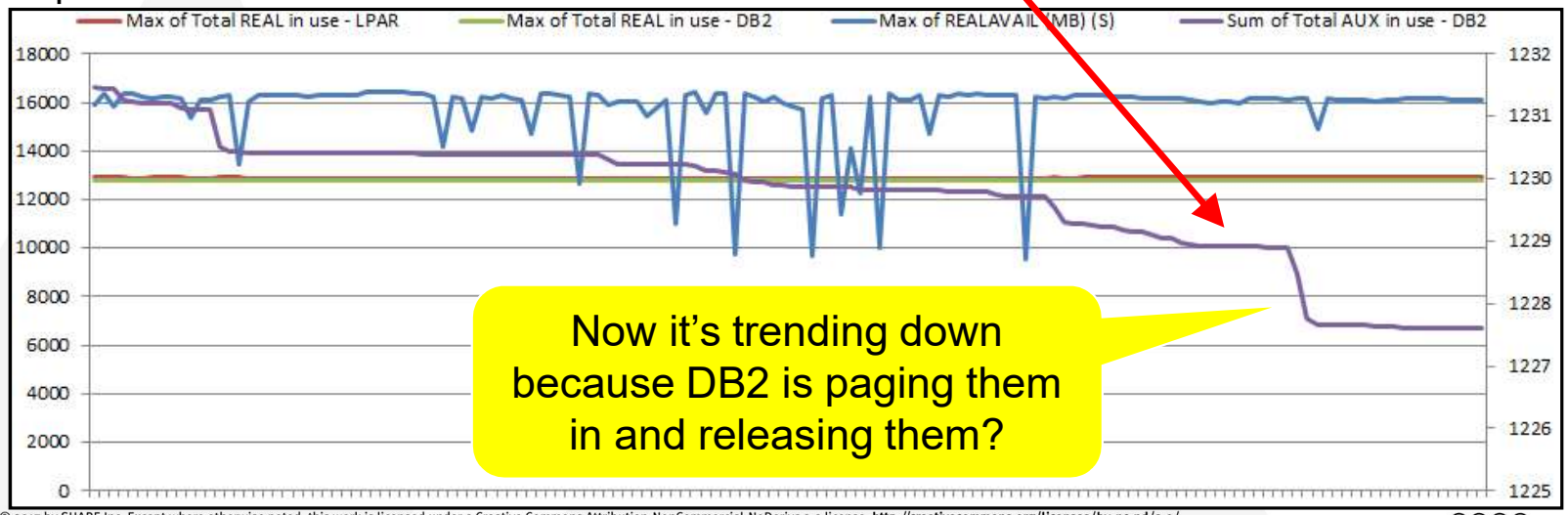


# Messages to be aware of…

- **AUX Storage**
  - Informational level — 50%
    - Issue ENF 55
    - Issue IRA205I: % of AUX is allocated
      - IRA265I if SCM installed
    - Also where DB2 goes into hard DISCARD mode
  - Warning level — 70%
    - Issue ENF 55
    - Issue IRA200E and IRA206I, or IRA260E (SCM): shortage of TOTAL AUX and list of top 20 consumers
    - Issue IRA203E swapping culprit out
    - Issue IRA210E set non-swappable AS as non-dispatchable (STORAGENSWDP=YES)
  - Critical Shortage level (everything seen in Shortage +) — 85%
    - Issue IRA201E Critical paging shortage
    - If this case continues for 15 seconds issue IRA220I and IRA221D to show largest consumers and allow operator to cancel them

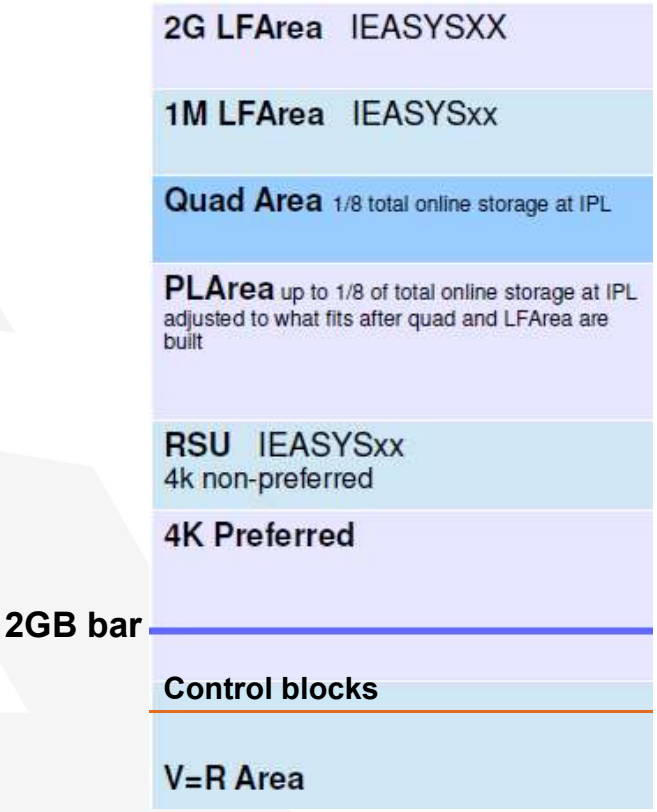
# More on AUX storage

- If you run out of AUX storage (page data sets full) z/OS will come down**
- AUX storage has been referred to as ‘double accounting’
- Once the page data sets (AUX slots) are utilized by an address space they remain assigned to that address space
  - UNTIL ‘DB2’ is bounced, ***or we page them in and release the AUX slot***… it looks like we are using it..
    - We have a requirement out to z/OS to address this



# Pieces of REAL storage (prior to z/OS V2.3)

- REAL storage in z/OS is categorized based on usage
  - 2GB LFAREA defined in IEASYSxx by user
  - 1MB LFAREA defined in IEASYSxx by user
    - Use INCLUDE1MAFC parm
  - QUAD area is made up of 4-chained 4k frames used for DAT translation
    - On z/OS 2.2 apply OA54945 so these frames can be used for other purposes
  - PLArea is pageable large frame area (12.5% of LPAR)
  - RSU is reconfigurable storage under guise of PR/SM
  - 4k Preferred means depending on the type of storage request this is where the 'best' 4k candidates are
    - Preferred might be Db2, non-preferred means swappable
  - Finally 1/64th of REAL is reserved for RSM control blocks



# Pieces of REAL storage…

- Example of how z/OS reserves real storage (**prior to z/OS V2.3)**
  - Starting position

| | | |
|---|---|---|
| Online memory (GB) | 100.0 | = What is allocated |
| LFAREA (GB) | 0 | = User defined |
| QUAD (GB) | 12.5 | = 12.5% of LPAR |
| 1MB PAGEABLE (GB) | 12.5 | = 12.5% of LPAR |
| RSM mapping | 1.5 | = ~1.5% of LPAR |
| 4KB FRAMES (GB) | 73.5 | = Balance left for 4K preferred |

  - If you were to add 100GB to the LPAR and define it all as LFAREA

| | |
|---|---|
| Online memory (GB) | 200.0 |
| LFAREA (GB) | 100.0 |
| QUAD (GB) | 25.0 |
| 1MB PAGEABLE (GB) | 25.0 |
| RSM mapping | 3.0 |
| 4KB FRAMES (GB) | 47.0 |

← Now you have fewer 4K frames to handle the 4K workload needs, including taking dumps quickly, without having to break down free 1M frames

Do not forget that Quad area and Pageable 1M area grow proportionally with additional REAL memory!

# ZPARM REALSTORAGE_MANAGEMENT= ??

- Options are AUTO (default), ON, OFF
  - It is DB2's attempt to be a good corporate citizen and free off 64-bit storage
- Affects when DB2 releases (discards) 64-bit storage
  - KEEPREAL(YES) means it still has DB2's name on it but RSM can reuse it, while KEEPREAL(NO) means the storage is completely freed
- RSM=OFF means No DISCARD unless REALSTORAGE_MAX hit OR 50% of AUX is in use
- "Soft discard" is DISCARD with KEEPREAL(YES)
  - RSM=AUTO **with no** paging means DISCARD with *KEEPREAL=YES* at Thread Deallocation or 120 commits
  - RSM=AUTO **with** paging means DISCARD with *KEEPREAL=YES* at Deallocation or 30 commits. STACK also DISCARDED
  - RSM=ON means DISCARD with *KEEPREAL=YES* at Deallocation or 30 commits. STACK also DISCARDED

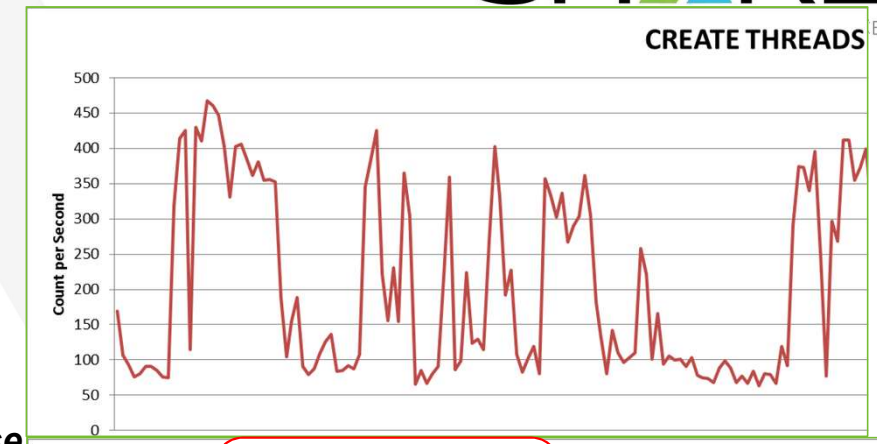
# Storage Contraction

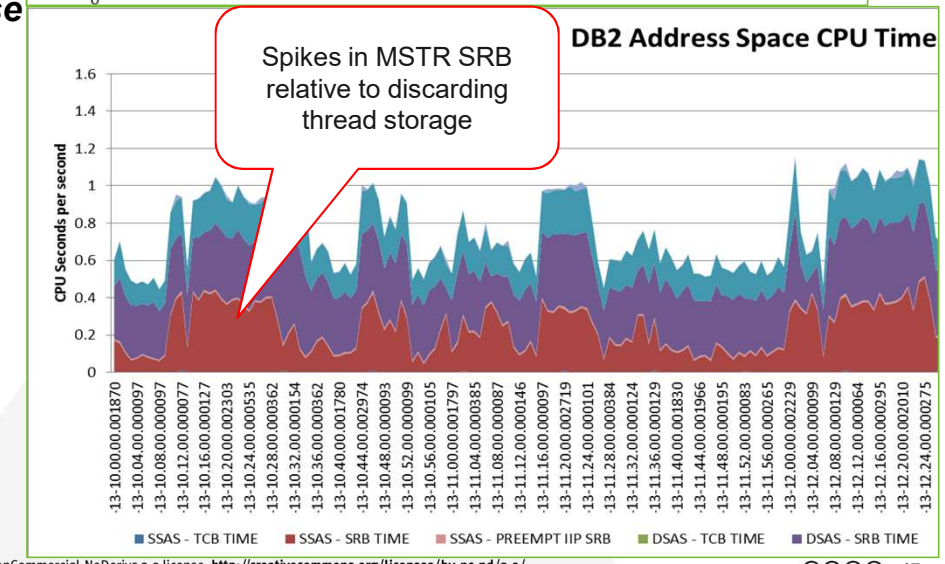
- "Hard discard" or Storage Contraction Mode is DISCARD with KEEPREAL(NO)
  - ENF 55 signal when 50% of AUX storage is in use means DISCARD KEEPREAL=NO
  - Hitting REALSTORAGE_MAX means DISCARD KEEPREAL=NO at 100%
  - We now report number of contractions in MEMU2 as well IFCID 001 (DISCARDMODE64 or RSMAX_WARN)
- Important notes:
  - Contraction mode is not exited immediately upon relief to avoid constant toggling in and out of this mode
- Contraction mode start is indicated by a DSNV516I message
  - **DSNVMON – BEGINNING STORAGE CONTRACTION MODE**
- Ending with a DSNV517I message
  - **DSNVMON – ENDING STORAGE CONTRACTION MODE**

- **With ample central storage (whitespace) and a cushion for MAXSPACE can turn RSM → OFF to save uncaptured and MSTR CPU time – savings directly correlates to frequency of thread deallocation**

# REALSTORAGE_MANAGEMENT

- RSM zparm controls when DB2 discards 64-bit storage (gives it back to z/OS) :: RSM = AUTO (default) → OFF
  - 64-bit Shared is almost all thread storage
- If there is ample storage on the LPAR investigate turning this off
  - Need to monitor DBM1 and HVSHARE storage on the LPAR to understand if there is storage growth after change
- Possible CPU savings ***are inversely proportional to thread reuse***
  - Poor thread reuse means more threads are deallocated and re-created constantly – and RSM needs to manage this
    - The release of storage from these deallocated threads costs cycles in the address space where the thread lives as well as MSTR
      - This customer saw 4x redux in MSTR SRB time during periods of high thread deallocation with RSM OFF
    - If POOLINAC constantly throws away threads DIST SRB could increase
    - With local attach threads DBM1 could be aggravated as well
    - There is also some CPU in z/OS RSM which is used to reclaim the storage



# Paging Public Service Announcement…

- What is a tolerable paging rate?
  - Only during dramatic workload shifts, hopefully that data will not be referenced again??
    - i.e. batch to online and vice versa?
  - This inherently means you have no ‘cushion’ for DUMPs, unexpected SORTing or Utility/Batch job which ran at an inopportune time….
- ….DB2’s answer is never
  - z/OS lab measured cost of I/O at 20us-70us of CPU
  - Don’t forget the elapsed time wasted as well (1-2ms avg.)
- Most customers have undersized dev/test environment….
  - **But at ~5us synch I/O (+1us no page fixing) and ~20us for prefetch (+5us for no page fixing) you are trading CPU for REAL memory
    - DB2 path length measurement
  - So why not right-size it to save CPU, and actually mirror the performance of what it will be in production??

# Potential DB2 Benefit from Larger Memory

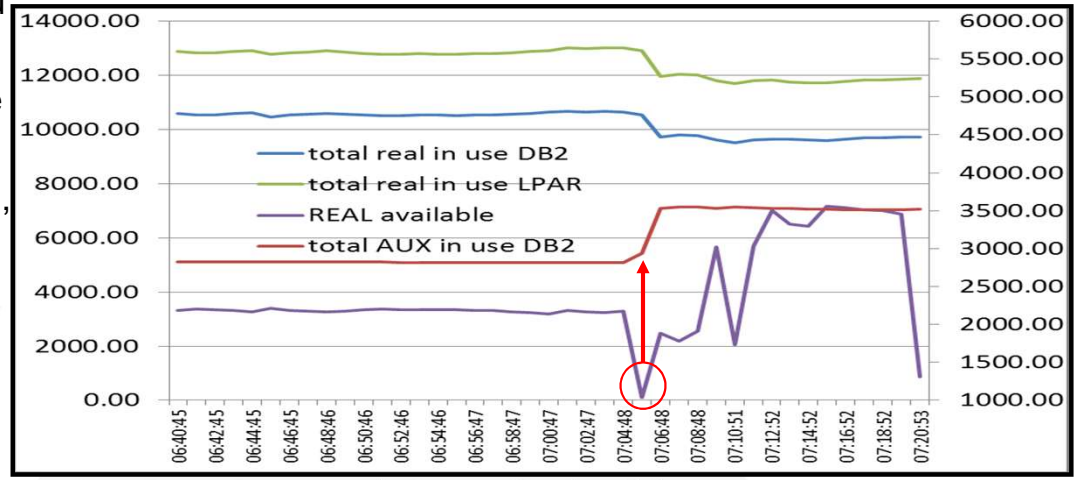
- Global dynamic statement cache
  - EDMSTMTC up to 4G with DB2 11, default 110MB
  - Reduction of CPU time by avoiding full prepare
- Local statement cache
  - MAXKEEPD up to 200K statements with DB2 11, default 5000
  - Reduction of CPU time by avoiding short prepare
- In-memory data cache for sparse index
  - MXDTCACH up to 512MB per thread, default 20MB
  - Reduction of CPU and elapsed time with potentially better access path selection with DB2 11
- RID Pool (MAXRBLK)
  - Lower CPU time if RID lists don't spill into work files
- SORTPOOL
  - Avoid spilling to a workfile
  - MAXSORT_IN_MEMORY - up to SORTPOOL for ORDER BY, GROUP BY
- DB2 local and group buffer pools
  - Reduction of elapsed time and CPU time by avoiding I/Os
  - PGSTEAL(NONE) in DB2 10 = In memory data base
  - CPU reduction from PGFIX=YES and large page frames
- Thread reuse with IMS or CICS applications
  - Reduction of CPU time by avoiding thread allocation and deallocation
- Thread reuse and RELEASE(DEALLOCATE)
  - Reduction of CPU time by avoiding package allocation and parent locks
  - DDF High performance DBATs support with DB2 10
  - Ability to break-in persistent thread with DB2 11
- V12 Enhancements!!

# DB2 12 and REAL storage

- DB2 12 will continue to trade real storage in return for CPU/zIIP savings
  - Thread footprint will likely increase 20%+ due to continuous delivery enhancements
- FTB or Fast Traversal Blocks – saw up to 45% getpage reduction
  - Goal is to cache all non-leaf pages in memory
  - Min of 10MB up to 200GB, default of 20% of allocated BPs
    - Control it with entries into SYSIBM.SYSINDEXCONTROL and ZPARM to disable is INDEX_MEMORY_CONTROL
    - -DISPLAY STATS(INDEXMEMORYUSAGE) - *DSNI070I -Db2G FAST INDEX TRAVERSAL STATUS: 66 MB, USED BY: 31 OBJECTS, CANDIDATE OBJECTS: 31*
  - Index requirements
    - Must be a unique index / Max of 64byte index entries
- Contiguous Buffer Pools - saw up to 8% CPU savings
  - Pre-req PGSTEAL(NONE)
    - Uses max of 10% or 6400 buffers for overflow area → increase VPSIZE by 6400 buffers
  - Eliminates LRU chains and LC14, LC24 contention

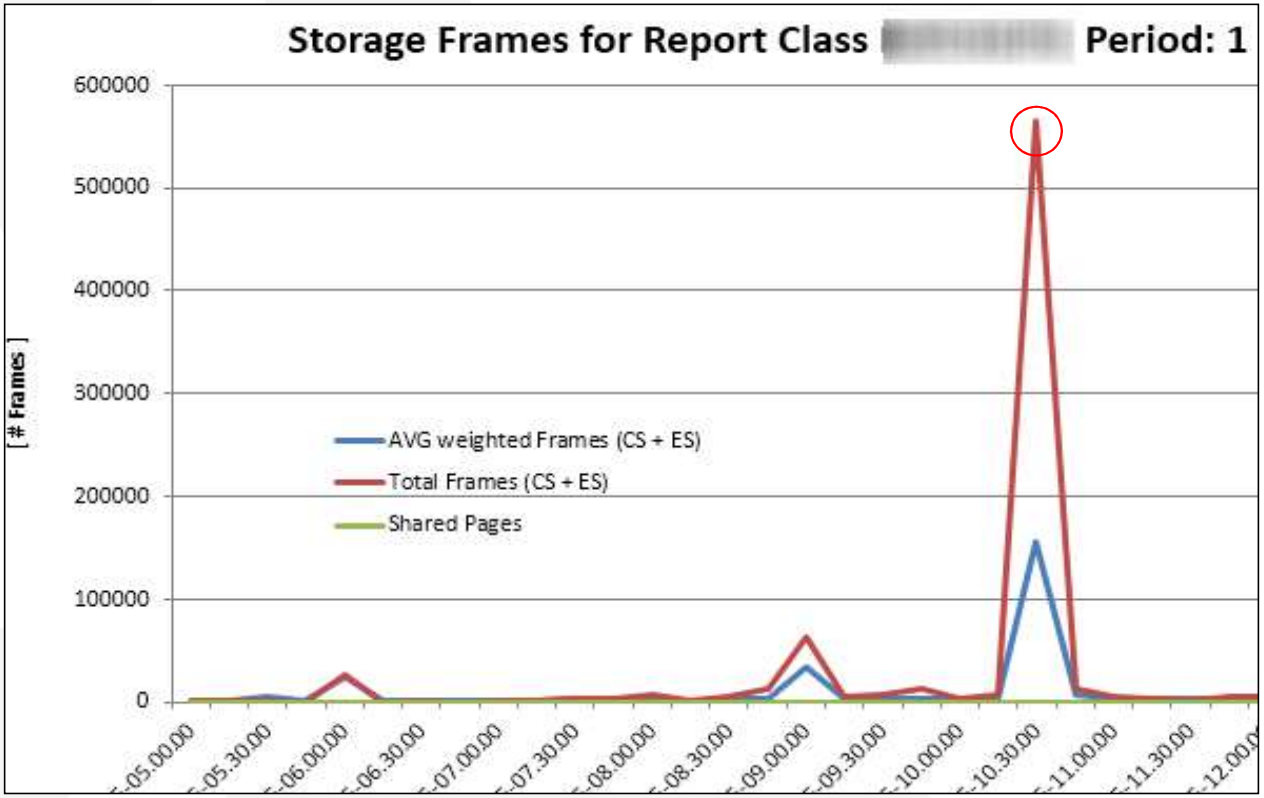
# DB2 Storage Shortage

- In the graphic we can see DB2 storage goes out from REAL to AUX when the real available drops to ‘0’ on the LPAR
- Worst case in this example to get those pages back in:
  - 700 MB – sync I/O time ~1ms = 0.002*179,200 = ~3 mins
  - If those pages were taken out of our buffer pools then we need to spend the I/Os to get the pages back in central storage – no prefetch from AUX
- Imagine a 16GB SVCDUMP occurring here!!
  - MAXSPACE = 16GB to allow for dump, should you reserve this space during peak processing hours?
  - MAXSNDSP = 15 seconds default (amount of time system non-dispatchable)
  - AUXMGMT=ON – no new dumps when AUX=50%, MAXSPACE always honored



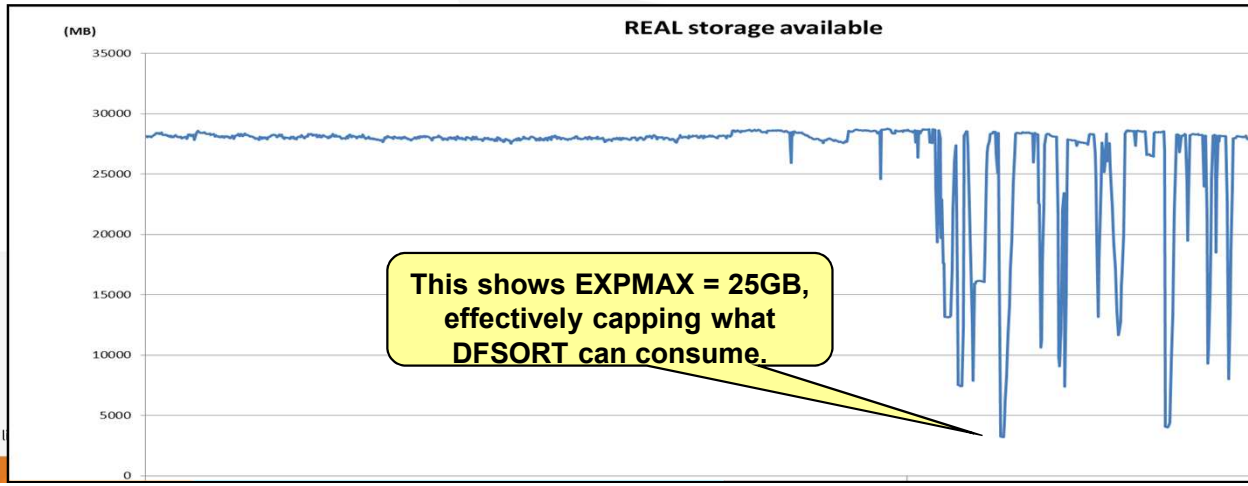
# DB2 Storage Shortage

- So who caused me to get paged out??
  - If you run a WLM activity report and look at the Storage Trend graph in the reporter you can see the actual frames used by a service or report class
  - The Page In Rates would also be high for that report class as data is brought in
  - Remember storage is paged out on an LRU basis, and one address space forces another out to AUX



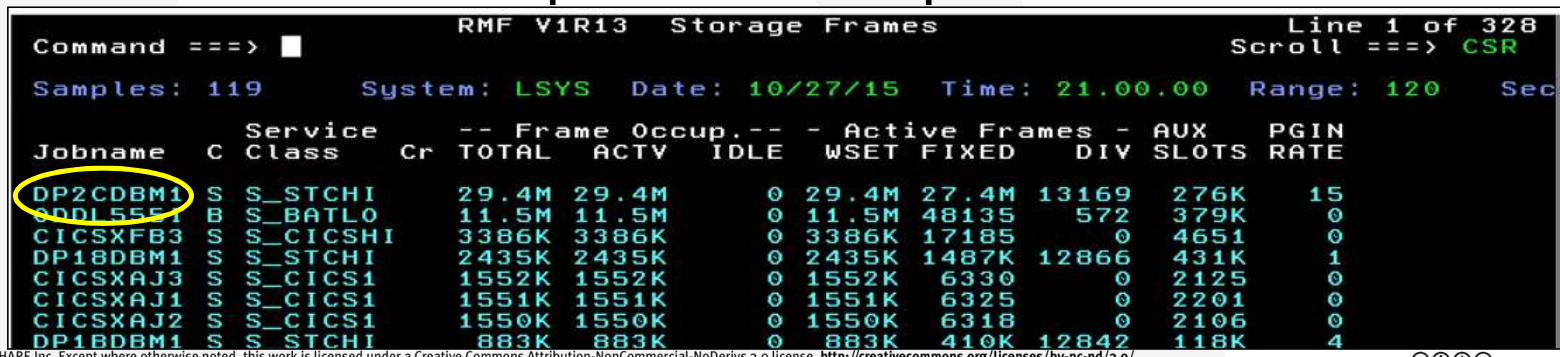
# Real storage and Sort products

- By default DFSORT and other sort products usually take as much storage as they can get, to help performance… but what about everyone else?
- DFSORT parameters affecting storage use (II13495 ) → a means to protect DB2
  - These can be dynamically changed for workloads using ICEPRMxx member
  - EXPOLD = % of storage allowed to be pushed to AUX → 0
  - EXPRES = % of storage to preserve, maybe in case of DUMPSPACE/ MAXSPACE → 16GB min in V10
  - EXPMAX = % of storage for memory object and hyperspace sorting, somewhat depends on EXPOLD and EXPRES → how much can you spare (**does not take into account page-fixed frames)**
- DB2Sort has:
  - PAGEMON=ON/OFF to limit central storage usage based on paging
  - PAGEMONL= total # of MB's of storage DB2SORT is allowed to consume



# Sort: 256GB LPAR ~114GB page fixed

- DB2 has over 1GB in AUX – page-in rate of 15 / second
- EXPMAX = 20% so SORT can use ~50GB max on the 256GB LPAR… BUT 114GB is PGFIXed!
  - Job using 45GB which is actually 31% of the available (256-114=142GB)
  - EXPMAX looks at total storage configured on the LPAR, regardless of PGFIX
- Look at EXPOLD → 0 and EXPMAX < *total storage - page fixed frames* to stop DFSORT from stealing old storage and pushing us out to AUX
- Ensure you do not end up with >70% of the LPAR page fixed
  - If 80% is fixed (IRA400E) and address spaces become non-dispatchable
- **WLM STORAGE CRITICAL = YES for DB2 address space could have helped here since the batch is a lower priority**



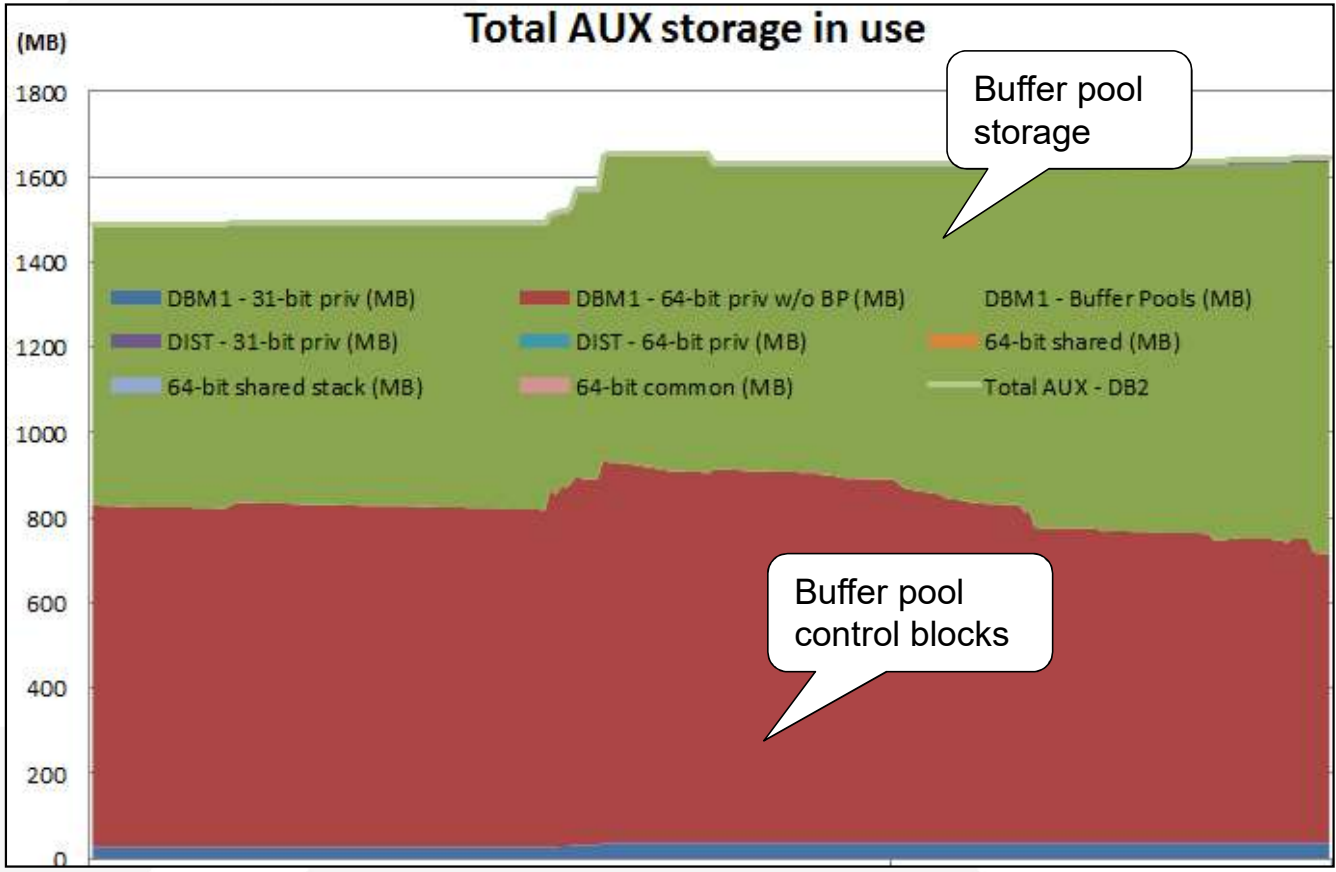
```
                          RMF V1R13  Storage Frames                 Line 1 of 328
Command ===>                                                      Scroll ===> CSR

Samples: 119      System: LSYS  Date: 10/27/15  Time: 21.00.00  Range: 120    Sec

                 Service      -- Frame Occup.--  - Active Frames -  AUX    PGIN
Jobname   C Class     Cr TOTAL   ACTV   IDLE   WSET FIXED    DIV SLOTS RATE
DP2CDBM1  S S_STCHI     29.4M 29.4M      0 29.4M 27.4M 13169  276K   15
DDDL555T  B S_BATLO     11.5M 11.5M      0 11.5M 48135   572  379K    0
CICSXFB3  S S_CICSHI    3386K 3386K      0 3386K 17185     0  4651    0
DP18DBM1  S S_STCHI     2435K 2435K      0 2435K 1487K 12866  431K    1
CICSXAJ3  S S_CICS1     1552K 1552K      0 1552K  6330     0  2125    0
CICSXAJ1  S S_CICS1     1551K 1551K      0 1551K  6325     0  2201    0
CICSXAJ2  S S_CICS1     1550K 1550K      0 1550K  6318     0  2106    0
DP1BDBM1  S S_STCHI      883K  883K      0  883K  410K 12842  118K    4
```

SHARE
EDUCATE ▸ NETWORK ▸ INFLUENCE

# Buffer Pools

# Bufferpools in AUX

- What shows up in DB2 statistics report regarding BP paged out to AUX?
  - Can I see evidence of paging in…
    - # of Sync I/O's?
    - BP Hit Ratio?
    - BP residency time?
  - **No! (BPs are Virtual)**
    - IFCID 225
    - Paging report
    - Class 2 not-accounted
    - z/OS uncaptured time
  - **Yes!**



# BP Statistics Example

- Prefetch Disabled – 3 possible reasons
  - No READ ENGINE – all 600 engines were in use so we need to speed up the prefetches, or spread the demand out (results in synch I/O sequential instead of prefetch)
  - NO BUFFER - **Either** because you hit the threshold of 90% of the buffers were 'dirty' **or** there were no sequential buffers allowed to handle the work (VPSEQT) – ROT: VPSIZE x VPSEQT≥320MB
    - VPPSEQT or VPSEQT = 0 and index I/O parallelism enabled at the zParm level
- PAGEINS for buffer pools
  - PAGEINS for READ/WRITE – DB2 came back to the oldest page on the LRU chain and in order to reclaim it for a new page we need to fix it in real before we read in the new page to take its place
    - This might occur at BP allocation, but should never be larger than the BP SIZE

If anything other than Pageins is > 0, increase the size of the virtual pool.

| BPNAME | DEFERED WRITE THRESHHOLD | PREFETCH DISABLED NO BUFFS | PAGINS FOR WRITE IO | PAGEINS FOR READ IO | CRITICAL THRESHOLD REACHED |
|---|---|---|---|---|---|
| BP4 | 0 | 14 | 0 | 19110 | 0 |
| BP4 | 0 | 103 | 0 | 0 | 0 |
| BP4 | 0 | 1 | 0 | 0 | |
| BP4 | 0 | 1 | 0 | 0 | |
| BP4 | 4 | 31 | 0 | 0 | |

BPSIZE=1000

# BP Simulation (V11 CM)

- If you have enough real…Try simulating changing VPSIZE and VPSEQT
  - Assumes LRU mechanism in place, can only simulate larger size
- Storage cost for a simulated buffer pool is ~2% for 4K pages similar to Hiperpool
  - SPSIZE * 4K (e.g. 20MB for 1GB size buffer pool)
  - Simulation is done with a separate buffer search hash table with buffer control blocks
  - CPU overhead is between 1-3% when running lab workloads
  - Shows up in DSNB431I-432I, and OMPE (IFCID 002)
- Look for law of diminishing returns
  - Track absolute number of I/Os saved per GB of buffers added to the pool
    - Use *-DIS BP(x) DETAIL over specific intervals*
  - When this number starts decreasing, move on to SPSEQT or the next buffer pool to tune
  - **Remember to increase GBP as well

*Does NOT take into account GBP cross-invalidation*

```
DSNB432I  -CEA1 SIMULATED BUFFER POOL ACTIVITY -
          AVOIDABLE READ I/O -
            SYNC  READ I/O (R)  =365071
            SYNC  READ I/O (S)  =5983
            ASYNC READ I/O      =21911
            SYNC  GBP READS (R) =89742
            SYNC  GBP READS (S) =184
            ASYNC GBP READS     =279
       PAGES MOVED INTO SIMULATED BUFFER POOL
=13610872
            TOTAL AVOIDABLE SYNC I/O DELAY =158014 MS
```

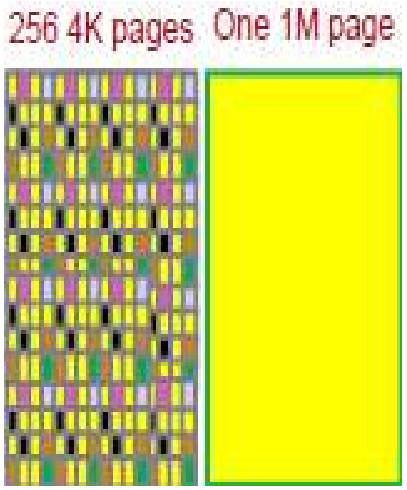
# BP Summary

- **Page fixing has highest benefit for pools with high I/O intensity**
- **Backing with larger frames has highest benefit for pools with high get page counts**
  - V12 adds GETPAGE counts to RTS
  - Potential for reduction of CPU through less TLB misses
  - CPU reduction based on customer experience 0 to 4%
  - Buffer pools must be defined as PGFIX=YES to use 1MB size page frames until V11
  - Must have sufficient total real storage to fully back the total DB2 requirement
- Involves partitioning real storage into 4KB and 1MB size page frames
  - Specified by LFAREA xx% or n in IEASYSnn parmlib member
  - Only changeable by IPL so estimate carefully
- If 1MB size page frames are overcommitted, DB2 will use 4KB page frames
  - Recommendation to add 5-10% to the size to allow for some growth and tuning
  - Must have both enough 4KB and enough 1MB size page frames

# Large Frames

# 1MB Frames

- TLB – translation lookaside buffer is a ‘fast-path’ through translation between virtual and REAL
  - Coverage today represents a much smaller fraction of an application’s working set size, including Db2 buffer pools leading to a larger number of TLB misses
  - Applications can suffer a significant performance penalty resulting from an increased number of TLB misses
- Solution:
  - Increase TLB coverage without proportionally enlarging the TLB size by using large pages
- Large Pages allow for a single TLB entry to fulfill many more address translations
- Analysis of TLB hit is done with Hardware Instrumentation Facility IFCID 113

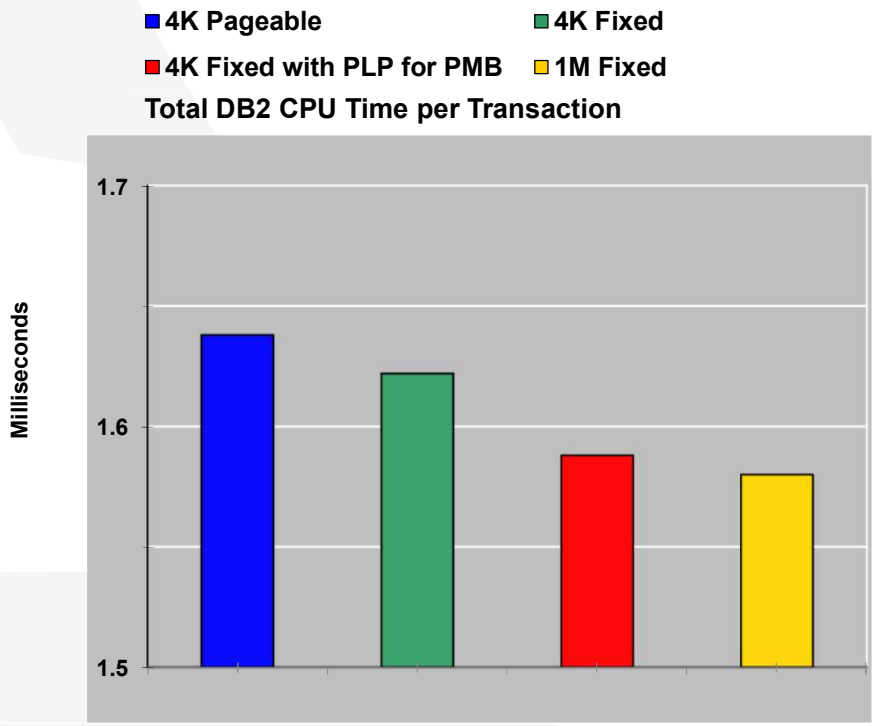



2GB Frame is 2,048 x larger than 1MB frame

```
94  TLB1MISS  NUM  8  6.1   TLB*CPU MISS*PERCENT OF*TOTAL CPU
96  PTEPCTMI  NUM  8  6.1   PAGETABLE*ENTRY*PCT OF TLB*MISSES
```

# Large Frames for BPs

- All of buffer pools are backed by real storage
- zEC12 16 CPs,  5000-6000 tps (mid to complex transactions) with 70 GB local buffer pools
- 120GB real storage with 75GB LFAREA configured for 1MB measurements
- 1MB frames with PGFIX=YES (long term page fix) is the best performer
- 4KB frames using PGFIX=YES and zEC12 Flash Express capability (1MB Pageable PMBs) is a good alternative if you cannot allocate LFAREA
  - Note : 70GB buffer pools are used, 8-10 sync I/O,  370 getpages per transaction



# 1MB Large Frames

- First exploited by DB2
  - Page fixed buffer pools in DB2 10
  - DB2 11 brings
    - 31-bit low private system storage
    - non-page fixed BP control blocks (PMBs) if it is FlashExpress compatible HW (also retrofit to DB2 10)
    - Ability to specify 1MB frame BPs without page fixing
    - OUTBUFF backed with 1MB frames
  - JAVA heap storage (controlled explicitly and monitored through traces)
- To allocate 1MB frames IEASYSxx in PARMLIB governs z/OS storage allocations
  - LFAREA=(1M=(*target*[%],*minimum*[%])
  - 2G=(*target*[%],*minimum*[%]))
    - We try to meet the targets, but if minimums cannot be met a console message is issued

# How do I size LFAREA for DB2 11?

SHARE
EDUCATE ▸ NETWORK ▸ INFLUENCE

- LFAREA = (1.04* (sum of VPSIZE from candidate buffer pools)) + 20MB + (OUTBUFF*1.15) + 31-bit low private

- **Do not oversize LFAREA**
  - LFAREA used ~ DB2 usage + JAVA heap (verbosegc traces)
  - Can't do anything about it until an IPL, if too small just means there is potential savings you could be missing out on
    - **- IRA127I 100% OF THE LARGE FRAME AREA IS ALLOCATED** = using it all
  - If for any reason RSM denies DB2 request for 1 MB frame, uses 4k instead
    - Specify INCLUDE1MAFC on LFAREA specification (OA42510)
- **Decomposing 1MB frames into 4k frames (due to paging w/out FlashExpress) is CPU intensive trying to maintain the LFAREA setting**

  - MAX LFAREA ALLOCATED (4K) = > 0 → Not a good sign

  - This indicates you do not have enough REAL storage needed by 4k frames, so add more real or make LFAREA smaller

# What about LFAREA?

- Useful commands
  - -DISPLAY BUFFERPOOL(ACTIVE) DETAIL(*)

    *DSNB546I -D2PW PREFERRED FRAME SIZE 2GB*

| FRAMESZ | BUFF4K | BUFF1M | BUFF2G |
|---|---|---|---|
| 2G | 327672 | 2440632 | 0 |
| 2G | 16344 | 25600 | 0 |
| 2G | 23128 | 107944 | 0 |

  - Why is BP not using large frames? Investigate z/OS side…
  - Useful command to find out how many 1MB size page frames are being used →
    - -DISPLAY VIRTSTOR,LFAREA

**NOT good, this means we broke down some of the 1MB frames**

**We reserve 1/8th of real on LPAR for pageable frames, then overflows to LFAREA**

```
IAR019I 14.37.22 DISPLAY VIRTSTOR 735
SOURCE =
TOTAL LFAREA = 4800M , 0G
LFAREA AVAILABLE = 42M , 0G
LFAREA ALLOCATED (1M) = 0M
LFAREA ALLOCATED (4K) = 4628M
MAX LFAREA ALLOCATED (1M) = 6M
MAX LFAREA ALLOCATED (4K) = 4703M
LFAREA ALLOCATED (PAGEABLE1M) = 130M
MAX LFAREA ALLOCATED (PAGEABLE1M) = 130M
LFAREA ALLOCATED NUMBER OF 2G PAGES = 0
MAX LFAREA ALLOCATED NUMBER OF 2G PAGES = 0
```

  - Answer: large frames are first come, first serve, must have enough REAL to back them
    - If using AUTOSIZE WLM will not always increment in 1MB or 2GB chunks**

# Pageable 1MB frames (prior to z/OS 2.3)

- By default z/OS 1.13 and up sets aside 1/8th of real storage on the LPAR for pageable large frames, as long as the RSM support for it is applied
  - See RMF paging report
- PMB buffer pool control blocks can use this
- Once the 1/8th is gone we need LFAREA to allocate the rest
  - *See previous slide*

| 01 MB Frames | Fixed | | | Pageable | | |
|---|---|---|---|---|---|---|
| | Total | Available | In-Use | Total | Available | In-Use |
| Min | 0 | . | 0 | 6,064 | 0 | 6,064 |
| Max | 0 | . | 0 | 6,064 | 0 | 6,064 |
| Avg | 0 | . | 0 | 6,064 | 0 | 6,064 |

- With SCM (FlashExpress) installed these frames can be paged out, but you lose any performance improvement those frames would have gotten from the TLB hit ratio

# New in z/OS 2.3

- LFAREA (1MB) managed by system with IEASYSxx *LFAREA* = acting as a cap as opposed to a target
  - 2GB LFAREA remains unchanged
- PLAREA or pageable area is now managed dynamically instead of being determined at IPL time, and is not capped

- New command to display large frame area
  - F AXR,IAXDMEM

```
IAR049I DISPLAY MEMORY V1.0
PAGEABLE 1M STATISTICS
      4824.0MB : TOTAL SIZE ← what RSM said it could use
      4585.0MB : AVAILABLE FOR PAGEABLE 1M PAGES
                 3.0MB : IN-USE FOR PAGEABLE 1M PAGES
                 3.0MB : MAX IN-USE FOR PAGEABLE 1M PAGES
                 1.0MB : FIXED PAGEABLE 1M FRAMES
LFAREA 1M STATISTICS - SOURCE = IEASYS23
      64.0MB : TOTAL SIZE ← the effective cap for 1MB frames
      62.0MB : AVAILABLE FOR FIXED 1M PAGES
                 2.0MB : IN-USE FOR FIXED 1M PAGES
                 2.0MB : MAX IN-USE FOR FIXED 1M PAGES
LFAREA 2G STATISTICS - SOURCE = IEASYS23
      0.0MB : TOTAL SIZE = 0 ← what is reserved for 2GB frames
      0.0MB : AVAILABLE FOR 2G PAGES = 0
                 0.0MB : IN-USE FOR 2G PAGES = 0
                 0.0MB : MAX IN-USE FOR 2G PAGES = 0
```

# FlashExpress

- FlashExpress is simply flash memory (storage class memory) for z196 and up machines
- Generally speaking for performance…
  - Access to real memory ~ 1,000 machine cycles
  - Access to FlashExpress ~ 100,000..
  - Access to AUX storage ~ 1,000,000..
- So is it a substitute for REAL?? - No
  - Never intended for buffer pools or working storage, it still pages, just in 1MB chunks
  - Better than AUX as an overflow for MAXSPACE (DUMPSERV)
- Highly recommended if CRITICAL PAGING is on (PPRC w/ HyperSwap fixes HVSHARE)
- Do not need it installed to get pageable 1MB frames – but if those frames are paged out they get broken down into 4k frames without SCM
- Performance numbers and a better description:
  - http://public.dhe.ibm.com/common/ssi/ecm/en/zss03073usen/ZSS03073USEN.PDF
  - 5x reduction in dump time, 3x reduction in non-dispatchable time compared to going out to AUX

# Summary – being short on REAL

- CPU cost of paging to AUX (DASD or FLASHEXPRESS)
  - z/OS measured 20-70us
- Elapsed time if sync I/O - Roughly 1-2ms with current DASD
  - Buffer pools, and other in-memory pools now moved to AUX
  - DB2 thread working storage pushed out
- DUMP capture can grind LPAR to a halt
- Other complications..
  - Lose positive impact if pageable 1MB frames, broken down or moved to SCM
  - Waste CPU breaking down LFAREA to 4k frames, and rebuilding them
  - If too many fixed frames (PGFIX or z/OS fixed frames) end up with storage critical situation and address spaces marked non-dispatchable

# Db2 SWAT team engagements

**Db2 Master Class**– held twice a year, one in the US and one in the UK

https://www.ibm.com/developerworks/community/wikis/home?lang=en#!/wiki/Wc05a3bbc003d_44bf_8673_d5dd7683d239/page/Db2%20for%20zOS%20Master%20Class%202019%20-%20Workshop%20Announcement

Hursley Lab – the week of 06/24

Silicon Valley Lab – San Jose the week of 09/23

Spend a week with John Campbell and the Db2 SWAT team covering performance and availability topics including how to analyze statistics and accounting data

**Db2 360 Degree Continuous Availability Assessment Study**

Comprehensive study performed by the Db2 SWAT team aimed at discovering exposures in continuous availability, performance, and speed of recovery

*Please contact me or Chunyang Xia (cxia@us.ibm.com)*

*It's all about robustness.*

# References

- Techdoc for V10 and V11 MEMU2 with spreadsheet sample
  - http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/PRS5279
- Subsystem and Transaction Monitoring and Tuning with DB2 11 for z/OS
  - http://www.redbooks.ibm.com/redpieces/abstracts/sg248182.html?Open
- RSM Enablement Offering (z/OS 1.13)
  - http://www-03.ibm.com/systems/z/os/zos/tools/downloads/
- Flash Express whitepaper
  - http://www-01.ibm.com/common/ssi/cgi-bin/ssialias?htmlfid=ZSS03073USEN&appname=wwwsearch
- Technote regarding REALSTORAGE_MANAGEMENT
  - http://www-01.ibm.com/support/docview.wss?uid=swg21971496
- OA50945 – Quad frame storage (1/8$^{th}$ of REAL) in z/OS 2.2 is non-stealable
  - https://www-304.ibm.com/support/entdocview.wss?uid=isg1OA50945

# Visit the SHARE Booth (#303)

SHARE
EDUCATE • NETWORK • INFLUENCE

## Educate

- Learn more about **hot topics** from peers in the **Tech Talk Corner**
- Take part in the **Hack-a-Thon** with IBM & Rocket

## Network

- **Engage** with fellow attendees
- Connect with the **zNextGen®** community
- Join the **#SHAREatl** social media conversation

## Influence

- Discover ways to **get involved** with SHARE
- Meet **SHARE Volunteers**

SHARE
EDUCATE ▸ NETWORK ▸ INFLUENCE

# Thank You for Attending!
# Please remember to complete your evaluation of this session in the SHARE mobile app.

## From VIRTUALly Constrained to REALly Overcommitted, 21146

Adrian Burke agburke@us.ibm.com