Save CPU and Improve Performance with z/OS Large Pages and Page Fixing

Dave Powell

DB2 Platform Engineer

June 21, 2016

IBM Z SYSTEMS CUSTOMER COUNCIL – VIENNA, VA

Agenda

- Operational Performance Objective
- Performance Improvements
  - Completed/Ongoing
  - Planned/In-Progress
- Requirements for Large Page Frames
- Benefits of Large Page Frames and Page Fixing
- Allocating Large Page Frames
- Sample Output –LFAREA and DB2 Bufferpool Display
- Test Environment – Engineering
- Test Case DB2 Table
- Procedure for Page Fixing DB2 Buffer Pools
- Test Cases
- Test Results
- So What ???????
- Suggestions for Testing 1MB Pages/Page Fixing
- Questions

Operational Performance Objective

- **Continue to provide the same, or better, application performance with the same, or fewer, mainframe resources (CPU, Memory, Disk) while supporting larger application workloads**

(Do more with less !!!!!!...and do it faster)

Performance Improvements - 1

- **Completed/Ongoing**
  - MVS
    - Turn on HiperDispatch
    - Turn on SMF Compression (DB2/CICS)
    - Reschedule Batch Work outside of Peak Online Window
    - Move workload from CPs to zIIPs
    - MWRT (Mobile Workload Reporting Tool)
    - WLM Service Class Tuning
    - Increased WLM Application Environments (MINSPAS)
  - DB2
    - Convert COBOL Stored Procedures to Native SQL (where appropriate)
      - 60% zIIP eligible
      - No WLM Application Environment STC needed
      - No EXCPs for application LOAD libraries
      - No Stored Procedure TCB scheduling waits
    - DDF Accounting Rollup (From 1 to 20 – 50)
    - Bind with RELEASE DEALLOCATE (big cpu savings in VSAM Transparency CICS)
    - Bind with DEGREE 1 (where appropriate)
    - Turning off DB2 Traces/IFCIDs to Omegamon and SMF
    - Reduce DB2 Parallelism Maximum DEGREE
    - Maximize CICS/DB2 Thread Reuse (protected/dedicated threads – DB2 Acctg=NONE)
    - Limit dynamic SQL CPU and Parallelism with RLF (Resource Limit Facility)
    - Minimize frequency of REORGs and RUNSTATs utilities

Performance Improvements - 2

- **Planned/In Progress**
  - Exploit SMT (Simultaneous Multi-Threading) for zIIP processors
  - Exploit Flashcopy for DB2 Image Copies
  - Exploit DB2 DDF HPDBATs (High Performance Database Access Threads)
  - Exploit Data Server Driver enableWLB=TRUE
  - Enable large page frames for DB2 Bufferpools (1MB/2GB pages) and Page-Fix

Requirements for Large Page Frames

- **Hardware-(2097 processor or above)**
  - z10(1MB), z196(1MB), EC12, z13
  - Flash Express Memory not Required for Page Fixed 1MB pages
- **Software**
  - zOS V1.9 and above
  - DB2 V10, V11 (DB2 V8/V9 supported bufferpool pagefixing)
- **System Changes**
  - Enable large pages thru update of IEASYSxx
    - Add LFArea=(%mm,xxM,xxG)
  - IPL LPAR
- **DB2**
  - V10-ALTER BPOOL(BPxx) PGFIX(YES) VPSIZE(xxxxx)
  - V11-ALTER BPOOL(BPxx) PGFIX(YES) FRAME(4K, 1M, 2G) VPMIN(xxxx) VPMAX(xxxxx)

Benefits of Large Page Frames and Page Fixing

- **Large Page Frame (1MB/2GB) Benefits**
  - Allows a single TLB (Translation Lookaside Buffer) to fulfill more address translations
  - Can minimize application performance penalty due to increased TLB misses
  - Can lower CPU from TLB hit
  - Recommended for DB2 Buffer Pools with highest number of Getpages

- **Page Fixing Benefits in DB2**
  - Avoid Page Fix and Page Free instructions for each
    - DB2 Read from Disk to Local Buffer Pool
    - DB2 Write to Disk from Local Buffer Pool
    - DB2 Write from Local Buffer Pool to Group Buffer Pool
    - DB2 Read from Group Buffer Pool to Local Buffer Pool
  - Page Fix instruction is now a more expensive 64bit instruction
  - Recommended for DB2 Buffer Pools with High I/O Intensity
    - I/O Intensity=(Pages Read + Pages Written) / (Number of buffer pages in pool)

Allocating Large Page Frames

- **Specified in IEASYSnn Parmlib Member**
  - LFAREA=(xxM,xxG)
  - ********** Only changeable by IPL **********
- **Useful Commands**
  - MVS Console
    - DISPLAY VIRTSTOR,LFAREA
  - DB2
    - -DISPLAY BUFFERPOOL(BPxx) SERVICE=4
      - Displays # of buffers satisfied by 4K and 1M pages
    - Need minimum of 6656 pages in Local Buffer Pool VPSIZE

Sample Output-LFAREA and DB2 Bufferpool Display

- **D VIRTSTOR,LFAREA**

```
IAR019I  12.30.08 DISPLAY VIRTSTOR 323
 SOURCE =  0E
 TOTAL LFAREA = 2048M , 0G
 LFAREA AVAILABLE = 1807M , 0G
 LFAREA ALLOCATED (1M) = 129M
 LFAREA ALLOCATED (4K) = 112M
 MAX LFAREA ALLOCATED (1M) = 234M
 MAX LFAREA ALLOCATED (4K) = 112M
 LFAREA ALLOCATED (PAGEABLE1M) = 0M
 MAX LFAREA ALLOCATED (PAGEABLE1M) = 0M
 LFAREA ALLOCATED NUMBER OF 2G PAGES = 0
 MAX LFAREA ALLOCATED NUMBER OF 2G PAGES = 0
```

- **-DISPLAY BUFFERPOOL(BP5) SERVICE=4**

```
DSNB402I  =DB2S BUFFER POOL SIZE = 20000 BUFFERS  AUTOSIZE = NO
          ALLOCATED      =    20000   TO BE DELETED   =       0
          IN-USE/UPDATED  =      33
DSNB406I  =DB2S PGFIX ATTRIBUTE -
          CURRENT = YES
          PENDING = YES
DSNB999I  =DB2S DSNB1DBP  SERVICE( 4 )OUTPUT
DSNB999I  =DB2S 4K PAGES 32
DSNB999I  =DB2S 1M PAGES 19968
```

Test Environment - Engineering

- **Hardware**
  - z13 - 2964 Model 606
  - CPs(2) (logical)
  - zIIPs(1) (logical)
  - 4GB memory
  - LFArea=2048M, 0G
  - Flash Memory
- **Software**
  - zOS V2.1
  - DB2 V10
- **DB2**
  - Tablespace Bufferpool VPSIZE=20,000
  - Index Bufferpool VPSIZE=20,000

Test Case DB2 Table

- **1 Partitioned Table**
  - 100 parts
  - Partitioned by Range
  - 50M Rows
  - Not Logged
- **1 Partitioned Index**
  - 100 parts
  - Non-Unique

Procedure for Page Fixing Buffer Pools

- **Issue STOP on DB2 Tablespace(s) and Index(s) in Buffer Pools**
- **Issue the following commands in order to move from non-page fixed to page-fixed DB2 Buffer Pools for each Buffer Pool:**
  - -ALTER BPOOL(BPx) PGFIX(YES)
    - PGFIX will be in Pending State (Current=NO, Pending=YES)
  - -ALTER BPOOL(BPx) VPSIZE(0)
    - Size=0, Allocated=0
  - -DISPLAY BPOOL(BPx) DETAIL SERVICE(4)
    - To verify Bufferpool pages are freed and allocated are 0
    - Verify PGFIX not Pending (Current=YES, Pending=YES)
  - -ALTER BPOOL(BPx) VPSIZE(xxxxx)
  - -DISPLAY BPOOL(BPx) DETAIL SERVICE(4)
    - Verify Size=xxxxx, Allocated=0
- **Issue START on DB2 Tablespace(s) and Index(s) in Buffer Pools**
- **Run large DB2 SQL SELECT to allocate pages**
- **-DISPLAY BPOOL(BPx) DETAIL SERVICE(4)**
  - Verify Allocated > 0, 4K Pages >=0, 1M Pages >=6656

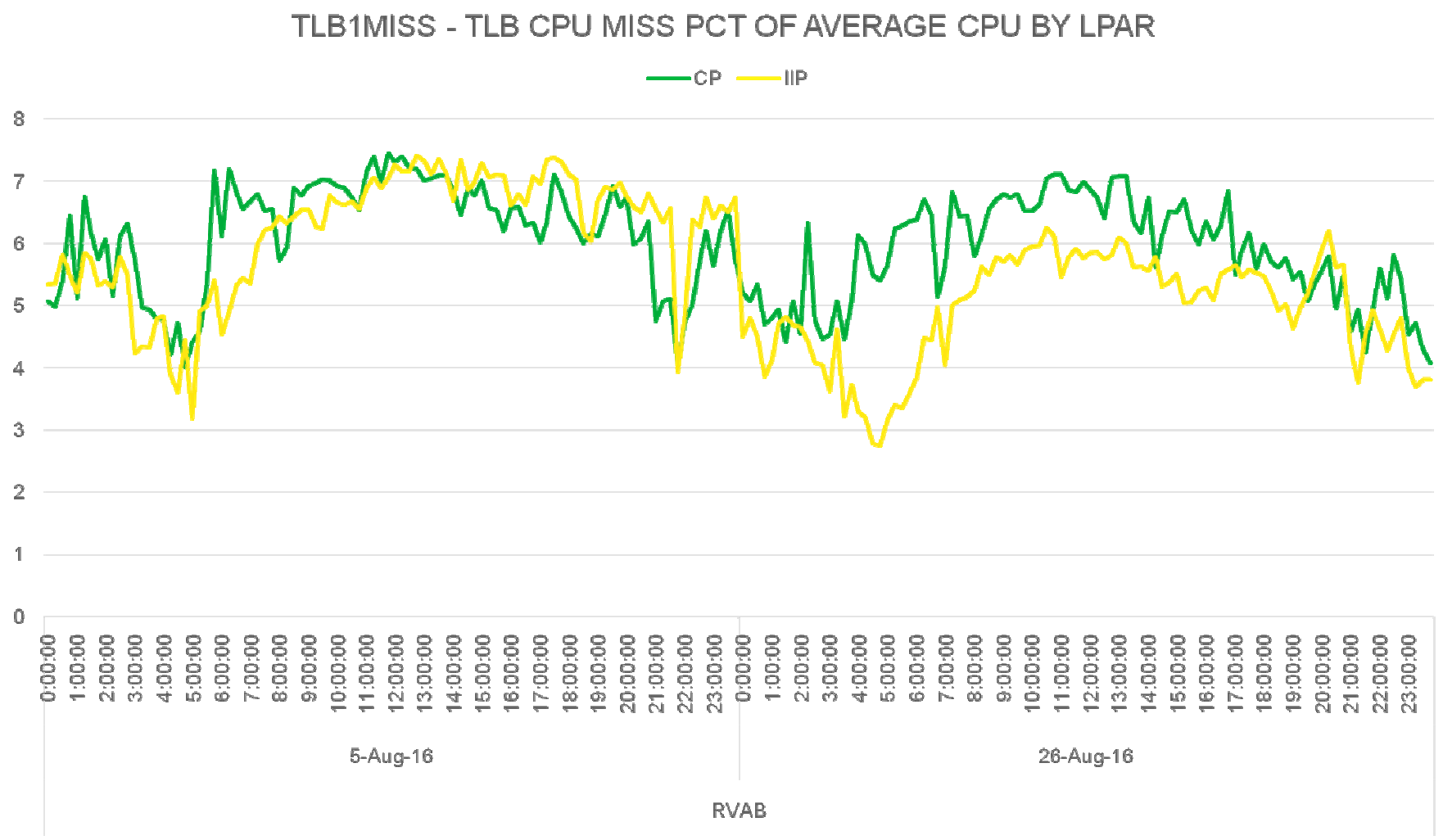
Test Cases

- **Test Case #1**
  - SELECT (no index access)
  - Table Scan
  - Dynamic and Sequential Prefetch
- **Test Case #2**
  - SELECT + UPDATE small # of rows (no index access/update)
  - Table Scan
  - Dynamic and Sequential Prefetch
- **Test Case #3**
  - UPDATE large number of rows (25M) (no index update)
  - Asynchronous Update
  - Dynamic and Sequential Prefetch
- **Test Case #4**
  - SELECT with Index Access
  - Synchronous Read (Index Only) and Dynamic Prefetch

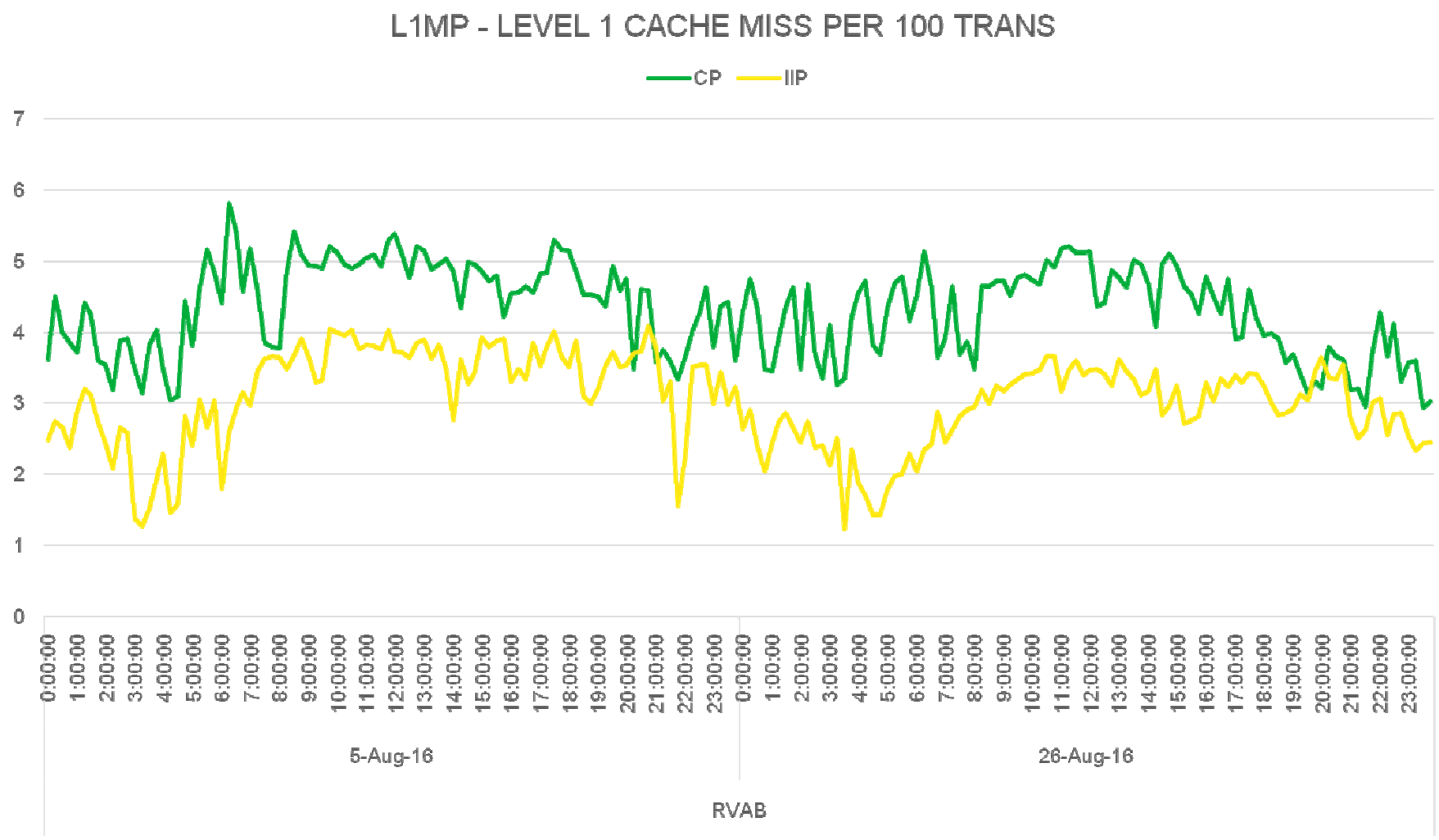
Test Results

- **Test Case #1**
  - No measurable consistent difference in elapsed or cpu time in Accounting (SMF 101)
  - 48% average lower DBM1 cpu time (most in Preemptable zIIP SRB time-SMF 101)
- **Test Case #2**
  - No measurable consistent difference in elapsed or cpu time in Accounting (SMF 101)
  - 49% average lower DBM1 cpu time (most in Preemptable zIIP SRB time-SMF 100)
- **Test Case #3**
  - No measurable consistent difference in elapsed or cpu time in Accounting (SMF 101)
  - 49% average lower DBM1 cpu time (most in Preemptable zIIP SRB time-SMF101)
- **Test Case #4**
  - Average reduction of 2% In-DB2 CPU and 3% reduction In-DB2 Elapsed time for DB2 Accounting Thread (SMF101)
  - 49% average lower DBM1 cpu time (time evenly split between Preemptable zIIP SRB and SRB time-SMF101)

TLB Misses Reports



Level 1 Cache Misses



So What ?????????????

- **Reduced zIIP time consumed in DBM1 means**
  - Less zIIP offload to CP for zIIP eligible work
  - More zIIP available for
    - DB2 Parallelism
    - Dynamic SQL coming in from DDF
    - Native SQL Stored Procedures
    - IBM DB2 Utilities
      - Runstats
      - Load/Reorg
    - Omegamon/DB2 NTH
    - Java processes
    - 3rd party products now exploiting zIIP processors

Suggestions for Testing 1MB Pages/Page Fixing

- **KISS (Keep It Simple ?^%$%$*)**
  - 1 Table, 1 Index
  - SET DEGREE to 1 to avoid parallelism zIIP/CP mix in comparing runs
  - 2 Buffer Pools (1-Table BP, 1-Index BP)
  - Keep local bufferpool size small to drive higher I/O (VPSIZE of 20,000 works well)
  - Prime Buffer Pools before test runs to allocate bufferpool storage and datasets
  - Run tests at “Quiet Time” for more accurate results
  - Run many tests over and over to achieve accurate results
  - DO NOT OVERALLOCATE LFAREA
    - 1MB/2GB pages can be demoted to 4K pages – Shredding Overhead
  - Periodically monitor LFAREA pages using console command:
    - D LFAREA
  - Watch out for other address spaces allocating 1MB Fixed and 1MB Pageable Storage
    - Omegamon MVS (64MB Fixed Memory Objects for Collector)
    - DB2 DBM1, zFS, OMVS, IXGLOGR, Java Heap Storage (1MB pageable storage),
  - If 1MB pageable pages get paged out when no Flash Express then demoted to 4K and never coalesced back to 1MB pages

Questions

??????????????????????????????

Appendix

References

- Robert's DB2 blog: DB2 10 for z/OS, 1 MB Page Frames, and the Number 6656, August 29, 2014
- SHARE Session 15142 March 2014 z Large Memory