Guide to Multiple Page Size Support on AIX 5L Version 5.3

March, 2006

David Hepkin

IBM Corporation

Contributors: Augie Mena, Greg Mewhinney

Introduction / Overview

The POWER5+™ processor and AIX 5L™ Version 5.3 with the 5300-04 Recommended Maintenance Package introduce support for two new virtual memory page sizes – 64KB and 16GB. Using larger virtual memory page sizes for an application's memory can significantly improve an application's performance and throughput due to hardware efficiencies associated with larger page sizes. While 16GB pages are intended to only be used in very high-performance environments, 64KB pages are general-purpose, and most workloads will likely see a benefit by using 64KB pages rather than 4KB pages. This document gives an overview of the new multiple page size support of AIX 5L and how a user can use these new page sizes for an application's memory to potentially improve performance.

Software Requirements

AIX 5L Version 5.3 with the 5300-04 Recommended Maintenance Package is required for 64KB and 16GB page size support.

The minimum IBM System p5™ microcode level required for 64KB and 16GB page size support is p5 System Release 240, Service Level 202.

In order to allocate 16GB huge pages on a system, Version 5 Release 2 of the Hardware Management Console (HMC) machine code is required.

Supported page sizes

AIX 5L Version 5.3 with the 5300-04 Recommended Maintenance Package supports up to four different page sizes, but the actual page sizes supported by a particular system will vary based on processor type. The following table lists the page sizes supported by AIX 5L Version 5.3 with the 5300-04 Recommended Maintenance Package and required System p™ hardware:

Page Size	Required Hardware	Requires User Configuration	Restricted	Kernel
4KB	ALL	No	No	64 & 32
64KB	IBM POWER5+™ or later	No	No	64 only
16MB	POWER4™ or later	Yes	Yes	64 & 32
16GB	POWER5+ or later	Yes	Yes	64 only

As with all previous versions of AIX® and AIX 5L, 4KB is the default page size for AIX 5L Version 5.3 with the 5300-04 Recommended Maintenance Package. A process will continue to use 4KB pages unless a user specifically requests another page size be used.

64KB Page Size Support

On POWER5+ systems, AIX 5L Version 5.3 with the 5300-04 Recommended Maintenance Package supports a new 64KB page size when running the 64-bit kernel. AIX 5L has rich support around 64KB pages, and 64KB pages are intended to be general-purpose. 64KB pages are very easy to use, and it is expected that many applications will see performance benefits when using 64KB pages rather than 4KB pages.

No system configuration changes are necessary to enable a system to use 64KB pages. On systems that support 64KB pages, the AIX 5L kernel will automatically configure 64KB pages for the system. 64KB pages are fully pageable, and the size of the pool of 64KB page frames on a system is dynamic and fully managed by AIX 5L. AIX 5L will vary the number of 4KB and 64KB page frames on a system to meet demand on the different page sizes. Both the **svmon** and **vmstat** commands can be used to monitor the number of 4KB and 64KB page frames on a system.

Page Sizes for Very High-Performance Environments

In addition to 4KB and 64KB, AIX 5L supports 16MB “large” pages and 16GB “huge” pages. These page sizes are intended to be used only in very high-performance environments, and AIX 5L will not automatically configure a system to use these page sizes. A system administrator must configure AIX 5L to use these page sizes and configure the number of pages of each of these page sizes. AIX 5L will not automatically change the number of configured 16MB or 16GB pages based on demand.

To configure the number of 16MB large pages on a system, a system administrator can use the **vmo** command. The following example configures 1GB of 16MB large pages:

```
# vmo -r -o lgpg_regions=64 -o lgpg_size=16777216
```

In the above example, the large page configuration changes will not take effect until the **bosboot** command is run and the system is rebooted. On systems that support dynamic logical partitioning (LPAR), the “*-r*” option can be omitted from the above command to dynamically configure 16MB large pages without a system reboot.

16GB huge pages must be configured via a system's Hardware Management Console (HMC). Under the "Managed System's Properties" menu, a system administrator can configure the number of 16GB huge pages on a system by selecting “Show Details” in the "Advanced Options" field of the "Memory" tab. Changing the number of 16GB huge pages on a system requires that the entire managed system be powered off.

Once a managed system has been configured with 16GB huge pages, a system administrator can assign 16GB huge pages to partitions by changing a partition's profile.

The memory allocated to 16MB pages can only be used for 16MB pages, and the memory allocated to 16GB pages can only be used for 16GB pages. Thus, pages of these

sizes should only be configured in high-performance environments. The use of 16MB and 16GB pages is restricted. In order to allocate pages of these sizes, a user must have the **CAP_BYPASS_RAC_VMM** and **CAP_PROPAGATE** capabilities or have root authority. The **chuser** command can be used to grant these capabilities to users.

Multiple Page Size Application Support

A user may specify page sizes to use for three regions of a 32-bit or 64-bit process's address space with an environment variable or with settings in an application's XCOFF binary via the **ldedit** or **ld** commands:

Region	**ld / ldedit option**	**LDR_CNTRL environment variable**	**Description**
Data	-bdatapsize	DATAPSIZE	Initialized data, bss, heap
Stack	-bstackpsize	STACKPSIZE	Initial thread stack
Text	-btextpsize	TEXTPSIZE	Main executable text

A user can specify a different page size to use for each of the three regions of a process's address space. For both interfaces, a page size should be specified in bytes. The specified page size may be qualified with a suffix to indicate the unit of the size. The supported suffixes are:

1. K - indicates kilobyte
2. M - indicates megabyte
3. G - indicates gigabyte

These may be specified in upper or lower case.

Only the 4KB and 64KB page sizes are supported for all three memory regions. The 16MB page size is only supported for the process data and process text regions. The 16GB page size is not supported for any of these regions.

If an unsupported page size is specified, the kernel will use the next smallest supported page size. If there is no page size smaller than the specified page size, the kernel will use 4KB pages.

Marking an application's binary with its preferred page sizes

A user can set an application's preferred page sizes in its XCOFF/XCOFF64 binary via the **ldedit** or **ld** commands. The **ld** or **cc** commands can be used to set these page size options when linking an executable:

```
ld –o mpsize.out –btextpsize:4K –bstackpsize:64K sub1.o sub2.o

cc –o mpsize.out –btextpsize:4K –bstackpsize:64K sub1.o sub2.o
```

The **ldedit** command can be used to set these page size options in an existing executable:

```
ldedit -btextpsize=4K -bdatapsize=64K -bstackpsize=64K mpsize.out
```

Note: The **ldedit** command requires the value for a page size option be specified with an equals sign ('='), but the **ld**/**cc** commands require the value for a page size option be specified with a colon (':').

Environment Variable

A user can also set a process's preferred page sizes via the **LDR_CNTRL** environment variable. For example, the following command will cause *mpsize.out* to use 4KB pages for its data, 64KB pages for its text, and 64KB pages for its stack on supported hardware:

```
$ LDR_CNTRL=DATAPSIZE=4K@TEXTPSIZE=64K@STACKPSIZE=64K mpsize.out
```

The page size environment variables override any page size settings in an executable's XCOFF header. Also, the **DATAPSIZE** environment variable will override any **LARGE_PAGE_DATA** environment variable setting.

Considerations

32-bit Processes
With the default AIX 5L 32-bit process address space model, a process's initial thread stack and data are located in the same PowerPC® 256MB segment. Currently, only one page size can be used in a segment. Thus, if different page sizes are specified for a standard 32-bit process's stack and data, the smaller page size will be used for both.

A 32-bit process can use different page sizes for its initial thread stack and data by using one of the alternate address space models for large and very large program support that locate a process's data heap in a segment other than its stack.

Thread Stacks
In a multi-threaded process, the stack page size setting will only apply to the stack for a process's initial thread. The thread stacks for subsequent threads are allocated from a process's data heap, and a process's data page size setting will be used for these stacks.

Using 64KB pages rather than 4KB pages for a multi-threaded process's data may reduce the maximum number of threads a process can create due to alignment requirements for stack guard pages. Applications that encounter this limit may disable stack guard pages by setting the environment variable **AIXTHREAD_GUARDPAGES** to 0.

Shared Libraries
On systems that support 64KB pages, AIX 5L will use 64KB pages for the global shared library text regions to improve performance.

Large Page Data

The **DATAPSIZE** environment variable will override the **LARGE_PAGE_DATA** environment variable. Also, the **datapsize** settings in an application's XCOFF binary will override any **lpdata** setting in the same binary.

Shared Memory

Because shared memory is not private to a process, there is no process-level environment variable or binary setting that selects a preferred page size for a process's shared memory. Instead, an application programmer can select the page size to use for System V shared memory via a new **SHM_PAGESIZE** command to the **shmctl()** system call:

```
#include <sys/shm.h>

int shmctl(int shmid, int command, struct shmid_ds *buffer);
```

When **SHM_PAGESIZE** is specified as the *command* parameter, **shmctl()** will set the page size backing the shared memory segment identified by *shmid* to the value of the *shm_pagesize* field of the *shmid_ds* structure specified by the *buffer* parameter. The *shm_pagesize* field is interpreted as a page size in bytes.

IMPORTANT: To change the page size backing a shared memory segment, **shmctl()** must be called with the **SHM_PAGESIZE** command immediately after the shared memory segment has been created and before any process has attached to the shared memory segment.

The **SHM_PAGESIZE** command can only be used by a process that has an effective user ID equal either to that of `superuser` or to the value of the `shm_perm.uid` or `shm_perm.cuid` fields in the *shmid_ds* data structure identified by the *shmid* parameter.

The **SHM_PAGESIZE** command can be used to allocate shared memory backed by any page size supported by the system. If an unsupported page size is specified in the *shm_pagesize* field, **shmctl()** will return -1 and set **errno** to *EINVAL*.

The **SHM_PAGESIZE** command is not supported on shared memory regions created by processes that use the **EXTSHM** environment variable.

The **SHM_PAGESIZE** command is not supported on pinned shared memory regions created with the **SHM_PIN** flag. However, after using the **SHM_PAGESIZE** command to set a shared memory region's page size, a process may pin the pages in a shared memory region by using the new **SHM_LOCK** command to **shmctl().** The **SHM_LOCK** command can only be used by a process that has an effective user ID equal to that of `superuser` or to the value of the `shm_perm.uid` or `shm_perm.cuid` field in the *shmid_ds* data structure identified by the *shmid* parameter. A non-superuser user must also have the **CAP_BYPASS_RAC_VMM** capability in order to use this command. Finally, the buffer parameter must be set to NULL when using the **SHM_LOCK** command.

When using a page size larger than 256MB, the minimum alignment of the address at which a shared memory region can be attached will be greater than 256MB (**SHMLBA**). To determine the minimum alignment at which a shared memory region can be attached, the **SHM_GETLBA** command can be specified to **shmctl()**. The **SHM_GETLBA** command will store the minimum alignment of a shared memory region in the *shm_lba* field of the *shmid_ds* struct pointed to by the *buffer* parameter. A process must have read permission for a shared memory region in order to use the **SHM_GETLBA** command.

The following example demonstrates how an application could allocate a 32MB shared memory region backed with 64KB pages:

```
int create_shm_region ()
{
        int    id;
        struct shmid_ds shm_buf = { 0 };
        size_t  shm_size  = 32*1024*1024;
        psize_t psize_64K = 64*1024;

        /* Allocate the shared memory region */
        if ((id = shmget(IPC_PRIVATE, shm_size, IPC_CREAT)) < 0)
        {
                perror("shmget() failed");
                return 1;
        }

        /* Use 64KB pages for the shared memory region */
        shm_buf.shm_pagesize = psize_64K;
        if (shmctl(id, SHM_PAGESIZE, &shm_buf))
        {
                perror("shmctl(SHM_PAGESIZE) failed");
                shmctl(id, IPC_RMID, NULL);
                return 2;

        }

        /* The shared memory region is backed with 64KB pages */
        return 0;
}
```

Gathering Page Size Information via User Commands

AIX 5L has added support to several commands to gather page size information.

Determining a System's Supported Page Sizes

To determine the page sizes supported by AIX 5L running on a particular system, the **"-a"** (**a**ll page sizes) and **"-f"** (**f**ormat) options to the **pagesize** command can be used:

```
# pagesize -af
4K
```

```
64K
16M
16G
```

Determining a Process's Page Sizes

The **ps** command can be used to monitor the page sizes being used for a process's data, stack, and text. To see a process's page sizes, the -Z flag can be specified to **ps**. With the -Z option, **ps** will display three new columns:

1. *DPGSZ* - process's data page size
2. *SPGSZ* - process's stack page size
3. *TPGSZ* - process's text page size

For example:

```
# ps -Z
    PID    TTY  TIME DPGSZ SPGSZ TPGSZ CMD
 311342  pts/4  0:00    4K    4K    4K ksh
 397526  pts/4  0:00    4K    4K    4K ps
 487558  pts/4  0:00   64K   64K    4K sleep
```

Monitoring Page Size Usage

vmstat

By default, the **vmstat** command displays memory statistics that are cumulative across all page sizes. To display memory statistics for a specific page size, the following **vmstat** options can be used:

vmstat option	Description
-p	Displays global vmstat information along with a break-down of statistics per page size
-P	Displays per page size statistics

Both options take a comma-separated list of specific page sizes or the keyword "**all**" to indicate information should be displayed for all supported page sizes that have one or more page frames. The following example displays per-page size information for all of the page sizes with page frames on a system:

```
# vmstat -P all

System configuration: mem=4096MB

pgsz            memory                           page
----- -------------------------- ------------------------------------
           siz      avm      fre    re    pi    po    fr      sr    cy
   4K   542846   202832   329649     0     0     0     0       0     0
  64K    31379      961    30484     0     0     0     0       0     0
```

svmon

The **svmon** command can be used to display page size usage across the system. Most **svmon** commands have been enhanced to provide a per-page size break-down of statistics. For example, to display global statistics about each page size, the "-G" command can be used to **svmon**:

```
# svmon -G
               size      inuse       free        pin    virtual
memory      8208384    5714226    2494158     453170    5674818
pg space     262144      20653

               work       pers       clnt
pin          453170          0          0
in use      5674818        110      39298

PageSize   PoolSize      inuse       pgsp        pin    virtual
s    4 KB          -    5379122      20653     380338    5339714
m   64 KB          -      20944          0       4552      20944
```

System Interfaces to Query Page Size Information

AIX 5L Version 5.3 with the 5300-04 Recommended Maintenance Package also provides an option to the **vmgetinfo()** system call and **kvmgetinfo()** kernel service for querying page size information:

```
int vmgetinfo(void *out, int cmd, int arg);
int kvmgetinfo(void *out, int cmd, int arg);
```

Querying Supported Page Sizes

The new **VMINFO_GETPSIZES** option to **vmgetinfo()** and **kvmgetinfo()** allows an application or kernel extension to query the page sizes supported by a system. When called with the **VMINFO_GETPSIZES** command and *arg* set to 0, **vmgetinfo()/kvmgetinfo()** will return the number of supported page sizes on the system.

When called with the **VMINFO_GETPSIZES** command and *arg* greater than 0, **vmgetinfo()/kvmgetinfo()** expect *out* to be a pointer to an array with *arg* number of *psize_t*'s. **vmgetinfo()/kvmgetinfo()** will fill the *out* array with the system's supported page sizes (in bytes) starting with the smallest supported page size, and it will return the number of page sizes set in the *out* array. Thus, the first element in the *out* array will correspond to the system's smallest supported page size.

The following example demonstrates how an application could determine a system's supported page sizes with **vmgetinfo()**:

```
int     num_psizes;
psize_t *psizes;

/* Determine the number of supported page sizes */
num_psizes = vmgetinfo(NULL, VMINFO_GETPSIZES, 0);

if (num_psizes < 0)
{
        /* This must be an older version of AIX that doesn't
         * support the VMINFO_GETPSIZES option.
         */
        perror("vmgetinfo() failed");
        return(1);
}

if ((psizes = malloc(num_psizes*sizeof(psize_t))) == NULL)
        return(2);

/* Get the page sizes */
if (vmgetinfo(psizes, VMINFO_GETPSIZES, num_psizes)
     != num_psizes)
{
        perror("vmgetinfo() failed");
        return(3);
}


/* psize[0] = smallest page size
 * psize[1] = next smallest page size...
 * psize[num_psizes-1] = largest supported page size
 */
```

Performance Considerations for Larger Page Sizes

The performance of certain applications may be noticeably affected by the use of 64KB, and larger, pages. For example, using 64KB pages for a variety of applications with measurable amounts of page translation overhead, including on-line transaction processing (OLTP) and Java™ workloads, has shown performance improvements in the range of 1% to 13% when compared to the default 4KB pages.

Factors to consider when trying to either estimate the potential benefit, or understand measured performance differences, of using larger pages instead of 4KB pages include: (1) memory usage, (2) a workload's page translation overhead.

The reason to consider memory usage is that for certain workloads, the use of a larger page size may result in memory fragmentation, which in turn could increase the memory footprint of an application. The **svmon** and **vmstat** commands can be used as described above to monitor memory usage. An increase in memory footprint could potentially result in a negative performance effect.

The main benefit of larger page sizes is improved performance for applications that repeatedly access large amounts of memory. Performance improvements from larger page sizes result from reducing the overhead of translating an application page address to a page address that is understood by the computer's memory subsystem. To improve performance, the information needed to translate a given page is usually cached in the processor. In POWER5+, this cache takes the form of a translation lookaside buffer (TLB). Since there are a limited number of TLB entries, using a large page size increases the amount of address space that can be accessed without incurring translation delays. For this reason, a measurement of the number of memory accesses that "miss" the TLB relative to the number of instructions executed, i.e., the "TLB miss rate", is an effective way to determine if there is a potential performance improvement in using a larger page size.

Obtaining the TLB miss rate can be complex. Given the user-friendly methods that have been provided for 64KB page size support, it is much easier to simply run a workload with 64KB page size enabled/disabled and measure the performance effects of using a larger page size.

While a detailed explanation of its use is beyond the scope of this paper, the **hpmcount** command can be used to count instructions executed and TLB misses while running a particular application. The **hpmcount** command is included in the **bos.pmapi.tools** fileset and is located at **/usr/pmapi/tools/hpmcount** when the fileset is installed. The following example would compute the TLB miss rate for the *mpsize_app* application on a POWER5+ machine:

```
hpmcount -s4 mpsize_app
```

Measuring the TLB miss rate is recommended only for those cases where it is critical that an application's page translation overhead characteristics be understood prior to deciding whether to try using a larger page size.

Conclusion

AIX 5L Version 5.3 with the 5300-04 Recommended Maintenance Package provides rich support for an application to utilize the different page sizes provided by the POWER5+ processor. Larger page sizes can be used for an application's memory without any application changes, allowing applications to easily utilize the performance efficiencies of larger page sizes like 64KB.

Additional Resources

For more information on the 5300-04 Recommend Maintenance Package for AIX 5L Version 5.3, see the AIX 5L Version 5.3 release notes:

http://publib.boulder.ibm.com/infocenter/pseries/v5r3/topic/com.ibm.aix.resources/53relnotes.htm

For more information on downloading and installing p5 System Release 240 microcode, see the microcode download page at:

http://techsupport.services.ibm.com/server/mdownload/home.html

For more information on using the **hpmcount** command, see the AIX 5L Version 5.3 documentation on **hpmcount**:

http://publib.boulder.ibm.com/infocenter/pseries/v5r3/topic/com.ibm.aix.doc/cmds/aixcmds2/hpmcount.htm

For more information on using AIX 5L, see the AIX 5L Version 5.3 Information Center:

http://publib.boulder.ibm.com/infocenter/pseries/v5r3/index.jsp?topic

© IBM Corporation 2006

IBM Corporation
Systems and Technology Group
Route 100
Somers, New York 10589

Produced in the United States of America
February 2006
All Rights Reserved

This document was developed for products and/or services offered in the United States. IBM may not offer the products, features, or services discussed in this document in other countries.

The information may be subject to change without notice. Consult your local IBM business contact for information on the products, features and services available in your area.

All statements regarding IBM future directions and intent are subject to change or withdrawal without notice and represent goals and objectives only.

IBM, the IBM logo, AIX, AIX 5L, POWER4, POWER5+, System p, are trademarks or registered trademarks of International Business Machines Corporation in the United States or other countries or both. A full list of U.S. trademarks owned by IBM may be found at: http://www.ibm.com/legal/copytrade.shtml.

Other company, product, and service names may be trademarks or service marks of others.

Information concerning non-IBM products was obtained from the suppliers of these products or other public sources. Questions on the capabilities of the non-IBM products should be addressed with those suppliers.

All performance information was determined in a controlled environment. Actual results may vary. Performance information is provided “AS IS” and no warranties or guarantees are expressed or implied by IBM. Buyers should consult other sources of information, including system benchmarks, to evaluate the performance of a system they are considering buying

When referring to storage capacity, 1TB equals total GB divided by 1000; accessible capacity may be less

The IBM home page on the Internet can be found at: http://www.ibm.com.

The IBM System p home page on the Internet can be found at: http://www.ibm.com/systems/p.