Supporting Multiple Page Sizes in the Solaris™ Operating System Appendix

Richard McDougall, PAE

Sun BluePrints™ OnLine—March 2004

http://www.sun.com/blueprints

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95045 U.S.A.
650 960-1300

Part No. 817-6242-10
Revision 1.0, 3/10/04
Edition: March 2004

Please
Recycle

Supporting Multiple Page Sizes in the Solaris™ Operating System Appendix

This article is a quick reference to the Solaris™ Operating System manual sections for the commands and library interfaces referenced the SunBluePrints article "Supporting Multiple Page Sizes in the Solaris Operating System." Reference this article for information about using the following new tools to perform tasks described in the referenced article:

- "cc(1)" on page 2
- "cpustat(1M)" on page 5
- "pmap(1)" on page 8
- "ppgsz(1)" on page 17
- "trapstat(1M)" on page 21
- "mpss.so.1(1)" on page 32
- "memcntl(2)" on page 36

cc(1)

```
NAME
     cc - C compiler
SYNOPSIS
     cc   [-#] [-###] [-Aname[(tokens)]] [-B[static|dynamic]]
          [-C] [-c] [-Dname[=tokens]] [-d[y|n]] [-dalign] [-E]
          [-errfmt[=[no%]error]] [-erroff[=t[,t...]]]
          [-errshort[=i]] [-errtags=a] [-errwarn[=t[,t...]]]
          [-fast] [-fd] [-flags] [-fnonstd] [-fns=[yes|no]]
          [-fprecision=p] [-fround=r] [-fsimple[=n]] [-fsingle]
          [-fstore] [-ftrap[=t[,t...]]]  [-G] [-g] [-H] [-hname]
          [-I[-|dir]] [-i] [-KPIC] [-Kpic] [-keeptmp] [-Ldir]
          [-lname] [-mc] [-misalign] [-misalign2] [-mr[,string]]
          [-mt] [-native] [-nofstore] [-O] [-ofilename] [-P] [-p]
          [-Q[y|n]] [-qp] [-Rdir[:dir...]]  [-S] [-s] [-Uname]
          [-V] [-v] [-Wc,arg] [-w] [-X[c|a|t|s]] [-x386] [-x486]
          [-xa] [-xalias_level[=a]] [-xarch=a] [-xautopar]
          [-xbuiltin[=a]] [-xCC] [-xc99[=o]] [-xcache=c] [-xcg89]
          [-xcg92] [-xchar[=o]] [-xchar_byte_order[=o]]
          [-xcheck=n] [-xchip=c] [-xcode=v] [-xcrossfile[=n]]
          [-xcsi] [-xdebugformat=[stabs|dwarf]]
          [-xdepend=[yes|no]] [-xdryrun] [-xe] [-xexplicitpar]
          [-xF] [-xhelp=f] [-xhwcprof=[enable|disable]] [-xild-
          off] [-xildon] [-xinline=[v[,v...]]]  [-xipo[=a]]
          [-xjobs=n] [-xldscope={v}] [-xlibmieee] [-xlibmil]
          [-xlic_lib=sunperf] [-xlicinfo] [-xlinkopt[=level]]
          [-xloopinfo] [-xM] [-xM1] [-xMerge] [-xmaxopt[=v]]
          [-xmemalign=ab] [-xnativeconnect[=a[,a]...]]  [-xnolib]
          [-xnolibmil] [-xOn] [-xopenmp[=i]] [-xP] [-xpagesize=n]
          [-xpagesize_heap=n] [-xpagesize_stack=n] [-xparallel]
          [-xpch=v] [-xpchstop] [-xpentium] [-xpg]
          [-xprefetch[=val[,val]]] [-xprefetch_level=l]
          [-xprofile=p] [-xprofile_ircache[=path]]
          [-xprofile_pathmap=collect_prefix:use_prefix] [-xreduc-
          tion] [-xregs=r[,r...]]  [-xrestrict[=f]] [-xs]
          [-xsafe=mem] [-xsb] [-xsbfast] [-xsfpconst] [-xspace]
          [-xstrconst] [-xtarget=t] [-xtemp=dir]
          [-xthreadvar[={o}[,o...]]  [-xtime] [-xtransition]
          [-xtrigraphs[=[yes|no]] [-xunroll=n]
          [-xustr={ascii_utf16_ushort|no}] [-xvector[={yes|no}]]
          [-xvis] [-xvpara] [-Yc,dir] [-YA,dir] [-YI,dir]
          [-YP,dir] [-YS,dir] [-Zll]
```

(*continued on next page*)

(*continued from preceding page)*

```
-xpagesize=n
          (SPARC) Set the preferred page size for the stack and
          the heap.

          The n value must be one of the following:
           8K 64K 512K 4M 32M 256M 2G 16G or default.
          You must specify a valid page size for the Solaris
          operating environment on the target platform, as
          returned by getpagesize(3C). If you do not specify a
          valid pagesize, the request is silently ignored at
          run-time. The Solaris environment offers no guarantee
          that the page size request will be honored.
          You can use pmap(1) or meminfo(2) to determine page
          size of the target platform.

          The -xpagesize option has no effect unless you use it
          at compile time and at link time.
          Note that this feature is not available on Solaris 7
          and 8. A program compiled with this option will not
          link on Solaris 7 and 8.

          If you specify -xpagesize=default, the Solaris environ-
          ment sets the page size. -xpagesize without an argument
          is the equivalent to -xpagesize=default.

          Compiling with this option has the same effect as set-
          ting the LD_PRELOAD environment variable to mpss.so.1
          with the equivalent options, or running the Solaris 9
          command ppgsz(1) with the equivalent options before
          running the program. See the Solaris 9 man pages for details.

          This option is a macro for -xpagesize_heap and
          -xpagesize_stack. These two options accept the same
          arguments as -xpagesize: 8K, 64K, 512K, 4M, 32M, 256M,
          2G, 16G, default. You can set them both with the same
          value by specifying -xpagesize=n or you can specify
          them individually with different values.

-xpagesize_heap=n
          (SPARC) Set the page size in memory for the heap. n can
          be 8K, 64K, 512K, 4M, 32M, 256M, 2G, 16G, or default.
          You must specify a valid page size for the Solaris
          operating environment on the target platform, as
          returned by getpagesize(3C). If you do not specify a
          valid page size, the request is silently ignored at run-time.

          You can use pmap(1) or meminfo(2) to determine page
          size at the target platform.
```

(*continued on next page)*

(*continued from preceding page)*

If you specify -xpagesize_heap=default, the Solaris environment sets the page size. -xpagesize_heap without an argument is the equivalent to -xpagesize_heap=default.

Compiling with this option has the same effect as setting the LD_PRELOAD environment variable to mpss.so.1 with the equivalent options, or running the Solaris 9 command ppgsz(1) with the equivalent options before running the program. See the Solaris 9 man pages for details.

Note that this feature is not available on Solaris 7 and 8. A program compiled with this option will not link on Solaris 7 and 8.

-xpagesize_stack=n

(SPARC) Set the page size in memory for the stack. n can be 8K, 64K, 512K, 4M, 32M, 256M, 2G, 16G, default. You must specify a valid page size for the Solaris operating environment on the target platform, as returned by getpagesize(3C). If you do not specify a valid page size, the request is silently ignored at run-time.

You can use pmap(1) or meminfo(2) to determine page size at the target platform.

If you specify -xpagesize_stack=default, the Solaris environment sets the page size. -xpagesize_stack without an argument is the equivalent to -xpagesize_stack=default.

Compiling with this option has the same effect as setting the LD_PRELOAD environment variable to mpss.so.1 with the equivalent options, or running the Solaris 9 command ppgsz(1) with the equivalent options before running the program. See the Solaris 9 man pages for details.

Note that this feature is not available on Solaris 7 and 8. A program compiled with this option will not link on Solaris 7 and 8.

cpustat(1M)

```
NAME
     cpustat - monitor  system  behavior  using  CPU  performance
     counters

SYNOPSIS
     cpustat -c eventspec [-c eventspec]... [-ntD] [ interval [count]]

     cpustat -h

DESCRIPTION
     The cpustat utility allows CPU performance  counters  to  be
     used to monitor the overall behavior of the CPUs in the system.

     If interval is specified,  cpustat  samples  activity  every
     interval  seconds,  repeating  forever. If a count is speci-
     fied, the statistics are repeated count  times.  If  neither
     are  specified,  an  interval  of  five seconds is used, and
     there is no limit to the number  of  samples  that  will  be
     taken.

OPTIONS
     The following options are supported:
     -c eventspec
           Specifies a set of  events  for  the  CPU  performance
           counters to monitor. The syntax of these event specif-
           ication can be determined using the -h option to cause
           the  usage  message  to be generated. The semantics of
           these event specifications can be determined by reading
           the  CPU  manufacturers  documentation  for  the events.
           See cpc_strtoevent(3CPC) for description  of the syntax.

           Multiple -c options may be specified,  in  which  case
           the  command  cycles  between the different event set-
           tings on each sample.

     -D    Enables debug mode.

     -h    Prints an extensive help message on  how  to  use  the
           utility  and  how  to  program the processor-dependent
           counters.

     -n    Omits all header output  (useful  if  cpustat  is  the
           beginning of a pipeline).

(continued on next page)
```

(*continued from preceding page)*

```
     -t    Prints an additional column of processor cycle counts,
           if available on the current architecture.

USAGE
     A closely related utility, cputrack(1), can be used to moni-
     tor  the  behavior of individual applications with little or
     no interference from other activities on the system.

     The cpustat utility must be run by the super-user, as  there
     is  an intrinsic conflict between the use of the CPU perfor-
     mance counters system-wide by cpustat and the use of the CPU
     performance  counters  to monitor an individual process (for
     example, by cputrack.)

     Once any instance of this utility has  started,  no  further
     per-process  or per-LWP use of the counters is allowed until
     the last instance of the utility terminates.

     The times printed by the command correspond to the wallclock
     time  when  the  hardware  counters  were  actually sampled,
     instead of when the program told the kernel to sample  them.
     The time is derived from the same timebase as gethrtime(3C).

     The processor cycle counts enabled by the -t  option  always
     apply  to both user and system modes, regardless of the set-
     tings applied to the performance counter registers.

     On some hardware platforms,  the  counters  are  implemented
     using  32-bit  registers. While the kernel attempts to catch
     all overflows to  synthesize  64-bit  counters,  because  of
     hardware  implementation restrictions, overflows may be lost
     unless the sampling  interval  is  kept  short  enough.  The
     events  most  prone  to  wrap are those that count processor
     clock cycles. If such an  event  is  of  interest,  sampling
     should  occur  frequently  so that less than 4 billion clock
     cycles can occur between samples.

     The output of cpustat is designed to be readily parseable by
     nawk(1)  and  perl(1), thereby allowing performance tools to
     be composed by embedding cpustat in scripts.  Alternatively,
     tools  may  be constructed directly using the same APIs that
     cpustat is built upon using the facilities of  libcpc(3LIB).
     See cpc(3CPC).

     The cpustat utility only monitors the CPUs that are accessi-
     ble  to  it  in  the  current  processor  set. Thus, several
     instances of the utility can be running on the CPUs in  dif-
     ferent  processor  sets. See psrset(1M) for more information
     about processor sets.
```

(*continued on next page)*

(*continued from preceding page*)

Because cpustat uses LWPs bound to CPUs, the utility may have to be terminated before the configuration of the relevant processor can be changed.

WARNINGS

By running the cpustat command, the super-user will forcibly invalidate all existing performance counter context. This may in turn cause all invocations of the cputrack command, and other users of performance counter context, to exit prematurely with unspecified errors.

If cpustat is invoked on a system that has CPU performance counters, but on which the packages containing the kernel support for those counters is not installed, the following message appears:

cpustat: CPU performance counters are inaccessible on this machine.

This error message implies that cpc_access() has failed and is documented in cpc_access(3CPC). Review this documentation for more information about the problem and possible solutions.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcpcu
Interface Stability	Evolving

SEE ALSO

cputrack(1), nawk(1), perl(1), iostat(1M), prstat(1M), psrset(1M), vmstat(1M), cpc(3CPC), cpc_access(3CPC), cpc_strtoevent(3CPC), gethrtime(3C), libcpc(3LIB), attributes(5)

Sun Microelectronics UltraSPARC I&II User’s Manual, January 1997, STP1031, http://www.sun.com/sparc

Intel Architecture Software Developer’s Manual, Volume 3: System Programmers Guide, 243192, http://developer.intel.com

pmap(1)

```
NAME
     pmap - display information about the address space of a pro-
     cess

SYNOPSIS
     /usr/bin/pmap [-rslF] [pid | core] ...

     /usr/bin/pmap -x [-aslF] [pid | core] ...

     /usr/bin/pmap -S [-alF] [pid | core] ...

DESCRIPTION
     The pmap utility prints information about the address  space
     of a process.

OPTIONS
     The following options are supported:

     -a    Prints anonymous and swap reservations for shared mappings.

     -F    Force. Grabs the target process even if  another  pro-
           cess has control.

     -l    Shows unresolved dynamic linker map names.

     -r    Prints the process's reserved addresses.

     -s    Prints HAT page size information.

USAGE
     The pmap utility prints information about the address  space
     of a process.

     Process Mappings

           /usr/bin/pmap [ -rslF ]  [ pid | core ] ...

           By default, pmap displays the mappings in the  virtual
           address  order  they  are mapped into the process. The
           mapping size, flags and mapped object name are shown.

(continued on next page)
```

(*continued from preceding page)*

Process anon/locked mapping details

/usr/bin/pmap -x [-aslF] [pid | core] ...

The -x option displays additional information per mapping. The size of each mapping, the amount of resident physical memory, the amount of anonymous memory, and the amount of memory locked is shown with this option. This does not include anonymous memory taken by kernel address space due to this process.

Swap Reservations

/usr/bin/pmap -S [-alF] [pid | core] ...
The -S option displays swap reservation information per mapping.

DISPLAY FORMATS

One line of output is printed for each mapping within the process, unless the -s option is specified, where one line is printed for a contiguous mapping of each hardware translation page size. The column headings are shown in parentheses below.

Virtual Address (Address)
The first column of output represents the starting virtual address of each mapping. Virtual addresses are displayed in ascending order.

Virtual Mapping Size (Kbytes)
The virtual size in kilobytes of each mapping.

Resident Physical Memory (RSS)
The amount of physical memory in kilobytes that is resident for each mapping, including that which is shared with other address spaces.

Anonymous Memory (Anon)
The number of pages, counted by using the system page size, of anonymous memory associated with the specified mapping. Anonymous memory shared with other address spaces is not included, unless the -a option is specified.

Anonymous memory is reported for the process heap, stack, for 'copy on write' pages with mappings mapped with MAP_PRIVATE (see mmap(2)).

(*continued on next page)*

```
(continued from preceding page)

    Locked (Locked)
           The number of pages locked within the mapping. Typical
           examples  are  memory locked with mlock() and System V
           shared memory created with SHM_SHARE_MMU.

     Permissions/Flags (Mode)
           The virtual memory permissions are shown for each map-
           ping. Valid permissions are:

           r:   The mapping may be read by the process.
           w:   The mapping may be written by the process.

User Commands                                              pmap(1)

           x:    Instructions that reside within the mapping  may
                 be executed by the process.

           Flags showing additional information for each  mapping
           may be displayed:

           s:    The mapping is shared such that changes made  in
                 the  observed address space are committed to the
                 mapped file, and  are  visible  from  all  other
                 processes sharing the mapping.

           R:    Swap space is not  reserved  for  this  mapping.
                 Mappings created with MAP_NORESERVE and System V
                 ISM shared memory mappings do not  reserve  swap
                 space.

     Mapping Name (Mapped File)
           A descriptive name for  each  mapping.  The  following
           major types of names are displayed for mappings:

              +o  A mapped file: For mappings  between  a  process
                 and a file, the pmap command attempts to resolve
                 the file name for each mapping. If the file name
                 cannot  be resolved, pmap displays the major and
                 minor number of the device containing the  file,
                 and the file system inode number of the file.

              +o  Anonymous memory: Memory  not  relating  to  any
                 named  object  or file within the file system is
                 reported as [ anon ].


(continued on next page)
```

(*continued from preceding page)*

The pmap command displays common names for certain known anonymous memory mappings, such as:

[heap]
The process heap.

[stack]
The process stack.

If the common name for the mapping is unknown, pmap displays [anon] as the mapping name.

+o System V Shared Memory: Mappings created using System V shared memory system calls are reported with the names shown below:

shmid=n:
The mapping is a System V shared memory mapping. The shared memory identifier that the mapping was created with is reported.

ism shmid=n:
The mapping is an "Intimate Shared Memory" variant of System V shared memory. ISM mappings are created with the SHM_SHARE_MMU flag set, in accordance with shmat(2) (see shmop(2)).

dism shmid=n:
The mapping is a pageable variant of ISM. Pageable ISM is created with the SHM_PAGEABLE flag set in accordance with shmat(2) (see shmop(2)).

+o Other: Mappings of other objects, including devices such as frame buffers. No mapping name is shown for other mapped objects.

Page Size (Pgsz)
The page size in kilobytes that is used for hardware address translation for this mapping. See memcntl(2) for further information.

(*continued on next page)*

(*continued from preceding page)*

```
     Swap Space (Swap)
           The amount of swap space in kilobytes that is reserved
           for this mapping. That is, swap space that is deducted
           from the total available pool of reservable swap space
           that  is  displayed  with  the  command  swap  -s. See
           swap(1M).

EXAMPLES
     Example 1: Displaying process mappings

     By default, pmap prints one line for each mapping within the
     address  space  of the target process. The following example
     displays the address space of a typical bourne shell:

          example$ pmap 102905
          102905:    sh
          00010000    192K r-x--  /usr/bin/ksh
          00040000      8K rwx--  /usr/bin/ksh
          00042000     40K rwx--    [ heap ]
          FF180000    664K r-x--  /usr/lib/libc.so.1
          FF236000     24K rwx--  /usr/lib/libc.so.1
          FF23C000      8K rwx--  /usr/lib/libc.so.1
          FF250000      8K rwx--    [ anon ]
          FF260000     16K r-x--  /usr/lib/en_US.ISO8859-1.so.2
          FF272000     16K rwx--  /usr/lib/en_US.ISO8859-1.so.2
          FF280000    560K r-x--  /usr/lib/libnsl.so.1
          FF31C000     32K rwx--  /usr/lib/libnsl.so.1
          FF324000     32K rwx--  /usr/lib/libnsl.so.1
          FF340000     16K r-x--  /usr/lib/libc_psr.so.1
          FF350000     16K r-x--  /usr/lib/libmp.so.2
          FF364000      8K rwx--  /usr/lib/libmp.so.2
          FF380000     40K r-x--  /usr/lib/libsocket.so.1
          FF39A000      8K rwx--  /usr/lib/libsocket.so.1
          FF3A0000      8K r-x--  /usr/lib/libdl.so.1
          FF3B0000      8K rwx--    [ anon ]
          FF3C0000    152K r-x--  /usr/lib/ld.so.1
          FF3F6000      8K rwx--  /usr/lib/ld.so.1
          FFBFC000     16K rw---    [ stack ]
           total     1880K

Example 2: Displaying memory allocation and mapping types

     The -x option can be used to provide information  about  the
     memory  allocation and mapping types per mapping. The amount
     of resident, non-shared  anonymous,  and  locked  memory  is
     shown for each mapping:
```

(*continued on next page)*

(*continued from preceding page*)

```
example$ pmap -x 102908
102908:   sh
Address   Kbytes     RSS    Anon  Locked Mode   Mapped File
00010000      88      88       -       - r-x--  sh
00036000       8       8       8       - rwx--  sh
00038000      16      16      16       - rwx--    [ heap ]
FF260000      16      16       -       - r-x--  en_US.ISO8859-1.so.2
FF272000      16      16       -       - rwx--  en_US.ISO8859-1.so.2
FF280000     664     624       -       - r-x--  libc.so.1
FF336000      32      32       8       - rwx--  libc.so.1
FF360000      16      16       -       - r-x--  libc_psr.so.1
FF380000      24      24       -       - r-x--  libgen.so.1
FF396000       8       8       -       - rwx--  libgen.so.1
FF3A0000       8       8       -       - r-x--  libdl.so.1
FF3B0000       8       8       8       - rwx--  [ anon ]
FF3C0000     152     152       -       - r-x--  ld.so.1
FF3F6000       8       8       8       - rwx--  ld.so.1
FFBFE000       8       8       8       - rw---  [ stack ]
--------   -----   -----   -----   ------
total Kb    1072    1032      56       -
```

The amount of incremental memory used by each additional instance of a process can be estimated by using the resident and anonymous memory counts of each mapping.

In the above example, the bourne shell has a resident memory size of 1032Kbytes. However, a large amount of the physical memory used by the shell is shared with other instances of shell. Another identical instance of the shell will share physical memory with the other shell where possible, and allocate anonymous memory for any non-shared portion. In the above example, each additional bourne shell uses approximately 56Kbytes of additional physical memory.

A more complex example shows the output format for a process containing different mapping types. In this example, the mappings are as follows:

```
0001000: Executable text, mapped from 'maps' program

0002000: Executable data, mapped from 'maps' program

0002200: Program heap

0300000: A mapped file, mapped MAP_SHARED
0400000: A mapped file, mapped MAP_PRIVATE

0500000: A mapped file, mapped MAP_PRIVATE | MAP_NORESERVE
```

(*continued on next page*)

(*continued from preceding page)*

```
        0600000: Anonymous memory, created by mapping /dev/zero

        0700000: Anonymous memory, created by mapping /dev/zero
                 with MAP_NORESERVE

        0800000: A DISM shared memory mapping created with SHM_PAGEABLE
                 with 8MB locked via mlock(2)

        0900000: A DISM shared memory mapping created with SHM_PAGEABLE
                 with 4MB of its pages touched.

        0A00000: A DISM shared memory mapping created with SHM_PAGEABLE
                 with none of its pages touched.

        0B00000: A ISM shared memory mapping created with SHM_SHARE_MMU

        example$ pmap -xs 15492
        15492:  ./maps
        Address  Kbytes     RSS    Anon  Locked Mode   Mapped File
        00010000       8       8       -       - r-x--  maps
        00020000       8       8       8       - rwx--  maps
        00022000   20344   16248   16248       - rwx--    [ heap ]
        03000000    1024    1024       -       - rw-s-  dev:0,2 ino:4628487
        04000000    1024    1024     512       - rw---  dev:0,2 ino:4628487
        05000000    1024    1024     512       - rw--R  dev:0,2 ino:4628487
        06000000    1024    1024    1024       - rw---    [ anon ]
        07000000     512     512     512       - rw--R [ anon ]
        08000000    8192    8192       -    8192 rwxs-    [ dism shmid=0x5]
        09000000    8192    4096       -       - rwxs-    [ dism shmid=0x4]
        0A000000    8192    8192       -    8192 rwxsR    [ ism shmid=0x2 ]
        0B000000    8192    8192       -    8192 rwxsR    [ ism shmid=0x3 ]
        FF280000     680     672       -       - r-x--  libc.so.1
        FF33A000      32      32      32       - rwx--  libc.so.1
        FF390000       8       8       -       - r-x--  libc_psr.so.1
        FF3A0000       8       8       -       - r-x--  libdl.so.1
        FF3B0000       8       8       8       - rwx--    [ anon ]
        FF3C0000     152     152       -       - r-x--  ld.so.1
        FF3F6000       8       8       8       - rwx--  ld.so.1
        FFBFA000      24      24      24       - rwx--    [ stack ]
        -------- ------- ------- ------- -------
        total Kb   50464   42264   18888   16384
```

(*continued on next page)*

(*continued from preceding page*)

Example 3: Displaying Page Size Information

The -s option can be used to display the hardware translation page sizes for each portion of the address space. (See memcntl(2) for futher information on Solaris multiple page size support).

In the example below, we can see that the majority of the mappings are using an 8K-Byte page size, while the heap is using a 4M-Byte page size.

Notice that non-contiguous regions of resident pages of the same page size are reported as separate mappings. In the example below, the libc.so library is reported as separate mappings, since only some of the libc.so text is resident:

```
example$ pmap -xs 15492
15492:  ./maps
Address  Kbytes     RSS    Anon  Locked Pgsz Mode   Mapped File
00010000       8       8       -       -   8K r-x--  maps
00020000       8       8       8       -   8K rwx--  maps
00022000    3960    3960    3960       -   8K rwx--    [ heap ]
00400000    8192    8192    8192       -   4M rwx--    [ heap ]
00C00000    4096       -       -       -    - rwx--    [ heap ]
01000000    4096    4096    4096       -   4M rwx--    [ heap ]
03000000    1024    1024       -       -   8K rw-s-  dev:0,2 ino:4628487
04000000     512     512     512       -   8K rw---  dev:0,2 ino:4628487
04080000     512     512       -       -    - rw---  dev:0,2 ino:4628487
05000000     512     512     512       -   8K rw--R  dev:0,2 ino:4628487
05080000     512     512       -       -    - rw--R  dev:0,2 ino:4628487
06000000    1024    1024    1024       -   8K rw---    [ anon ]
07000000     512     512     512       -   8K rw--R    [ anon ]
08000000    8192    8192       -    8192    - rwxs-    [ dism shmid=0x5]
09000000    4096    4096       -       -   8K rwxs-    [ dism shmid=0x4]
0A000000    4096       -       -       -    - rwxs-    [ dism shmid=0x2]
0B000000    8192    8192       -    8192   4M rwxsR    [ ism shmid=0x3 ]
FF280000     136     136       -       -   8K r-x--  libc.so.1
FF2A2000     120     120       -       -    - r-x--  libc.so.1
FF2C0000     128     128       -       -   8K r-x--  libc.so.1
FF2E0000     200     200       -       -    - r-x--  libc.so.1
FF312000      48      48       -       -   8K r-x--  libc.so.1
FF31E000      48      40       -       -    - r-x--  libc.so.1
FF33A000      32      32      32       -   8K rwx--  libc.so.1
FF390000       8       8       -       -   8K r-x--  libc_psr.so.1
FF3A0000       8       8       -       -   8K r-x--  libdl.so.1
FF3B0000       8       8       8       -   8K rwx--    [ anon ]
FF3C0000     152     152       -       -   8K r-x--  ld.so.1
FF3F6000       8       8       8       -   8K rwx--  ld.so.1
FFBFA000      24      24      24       -   8K rwx--    [ stack ]
-------- ------- ------- ------- -------
total Kb   50464   42264   18888   16384
```

(*continued on next page*)

(*continued from preceding page)*

```
Example 4: Displaying swap reservations
     The -S option can be used to describe the swap  reservations
     for  a  process.  The  amount  of  swap  space  reserved  is
     displayed for each mapping within the process. Swap reserva-
     tions  are  reported as zero for shared mappings, since they
     are accounted for only once system wide.
             example$ pmap -S 15492
             15492:  ./maps
              Address  Kbytes    Swap Mode   Mapped File
             00010000       8       - r-x--  maps
             00020000       8       8 rwx--  maps
             00022000   20344   20344 rwx--    [ heap ]
             03000000    1024       - rw-s-  dev:0,2 ino:4628487
             04000000    1024    1024 rw---  dev:0,2 ino:4628487
             05000000    1024     512 rw--R  dev:0,2 ino:4628487
             06000000    1024    1024 rw---    [ anon ]
             07000000     512     512 rw--R    [ anon ]
             08000000    8192       - rwxs-    [ dism shmid=0x5]
             09000000    8192       - rwxs-    [ dism shmid=0x4]
             0A000000    8192       - rwxs-    [ dism shmid=0x2]
             0B000000    8192       - rwxsR    [ ism shmid=0x3]
             FF280000     680       - r-x--  libc.so.1
             FF33A000      32      32 rwx--  libc.so.1
             FF390000       8       - r-x--  libc_psr.so.1
             FF3A0000       8       - r-x--  libdl.so.1
             FF3B0000       8       8 rwx--    [ anon ]
             FF3C0000     152       - r-x--  ld.so.1
             FF3F6000       8       8 rwx--  ld.so.1
             FFBFA000      24      24 rwx--    [ stack ]
             -------- ------- -------
             total Kb   50464   23496

     The swap reservation information can be used to estimate the
     amount of virtual swap used by each additional process. Each
     process consumes virtual swap from  a  global  virtual  swap
     pool.  Global  swap reservations are reported by the 'avail'
     field of the swap(1M) command.

EXIT STATUS
     The following exit values are returned:
     0     Successful operation.
     non-zero
           An error has occurred.
FILES
     /proc/*
           process files
     /usr/proc/lib/*
           proc tools supporting files
```

(*continued on next page)*

(*continued from preceding page*)

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWesu (32-bit)
	SUNWesxu (64-bit)
Interface Stability	
Command Syntax	Evolving
Output Format(s)	Unstable

SEE ALSO

ldd(1), mdb(1), proc(1), ps(1), swap(1M), mmap(2), memcntl(2), shmop(2), dlopen(3DL), proc(4), attributes(5)

ppgsz(1)

NAME

ppgsz - set preferred stack and/or heap page size

SYNOPSIS

```
/usr/bin/ppgsz [-F] -o option[,option]  cmd | -p pid...
```

DESCRIPTION

The ppgsz utility sets the preferred stack and/or heap page size for the target process(es), that is, the launched cmd or the process(es) in the pid list. ppgsz stops the target process(es) while changing the page size. See memcntl(2).

(*continued on next page*)

```
(continued from preceding page)

OPTIONS
     The following options are supported:

     -F    Force. Sets the preferred page size options(s) for
           target process(es) even if controlled by other
           process(es). Caution should be exercised when using
           the -F flag. See proc(1).

     -p pid
           Sets the preferred page size option(s) for the target
           process(es) in the process-id (pid) list following the
           -p option. The pid list can also consist of names in
           the /proc directory. Only the process owner or the
           super-user is permitted to set page size.

           cmd is interpreted if -p is not specified. ppgsz
           launches cmd and applies page size option(s) to the
           new process.

           The heap and stack preferred page sizes are inherited.
           Child process(es) created (see fork(2)) from the
           launched process or the target process(es) in the pid
           list after ppgsz completes will inherit the preferred
           heap and stack page sizes. The preferred page sizes
           are set back to the default system page size on
           exec(2) (see getpagesize(3C)).

     -o option[,option]
           The options are:
           heap=size
                 This option specifies the preferred page size
                 for the heap of the target process(es). heap is
                 defined to be the bss (uninitialized data) and
                 the brk area that immediately follows the bss
                 (see brk(2)). The preferred heap page size is
                 set for the existing heap and for any additional
                 heap memory allocated in the future. See NOTES.

           stack=size
                 This option specifies the preferred page size
                 for the stack of the target process(es). The
                 preferred stack page size is set for the exist-
                 ing stack and newly allocated parts of the stack
                 as it expands.

(continued on next page)
```

(*continued from preceding page)*

At least one of the above options must be specified.

size must be a supported page size (see pagesize(1)) or 0, in which case the system will select an appropriate page size (see memcntl(2)).

size defaults to bytes and can be specified in octal (0), decimal, or hexadecimal (0x). The numeric value can be qualified with K, M, G, or T to specify Kilobytes, Megabytes, Gigabytes, or Terabytes, respectively. 4194304, 0x400000, 4096K, 0x1000K, and 4M are different ways to specify 4 Megabytes.

EXAMPLES

Example 1: Setting the preferred heap and stack page size

The following example sets the preferred heap page size to 4M and the preferred stack page size to 512K for all ora-owned processes running commands that begin with ora:

```
example% ppgsz -o heap=4M,stack=512K -p ‘pgrep -u ora ’^ora’’
```

EXIT STATUS

If cmd is specified and successfully invoked (see exec(2)), the exit status of ppgsz will be the exit status of cmd. Otherwise, ppgsz will exit with one of the following values:

0 Successfully set preferred page size(s) for processes in the pid list.

125 An error occurred in ppgsz. Errors include: invalid argument, invalid page size(s) specified, and failure to set preferred page size(s) for one or more processes in the pid list or cmd.

126 cmd was found but could not be invoked.

127 cmd could not be found.

FILES

/proc/*
Process files.

/usr/lib/ld/map.bssalign
A template link-editor mapfile for aligning bss (see NOTES)

(*continued on next page)*

(*continued from preceding page)*

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWesu (32-bit)
	SUNWesxu (64-bit)
Interface Stability	Evolving

SEE ALSO

ld(1), mpss.so.1(1), pagesize(1), pgrep(1), pmap(1), proc(1), brk(2), exec(2), fork(2), memcntl(2), sbrk(2), getpagesize(3C), proc(4), attributes(5)

Linker and Libraries Guide

NOTES

Due to resource constraints, the setting of the preferred page size does not necessarily guarantee that the target process(es) will get the preferred page size. Use pmap(1) to view the actual heap and stack page sizes of the target process(es) (see pmap -s option).

Large pages are required to be mapped at addresses that are multiples of the size of the large page. Given that the heap is typically not large page aligned, the starting portions of the heap (below the first large page aligned address) are mapped with the system memory page size. See getpagesize(3C).

To provide a heap that will be mapped with a large page size, an application can be built using a link-editor (ld(1)) mapfile containing the bss segment declaration directive. Refer to the section ‘‘Mapfile Option’’ in the Linker and Libraries Guide for more details of this directive and the template mapfile provided in /usr/lib/ld/map.bssalign. Users are cautioned that an alignment specification may be machine-specific and may lose its benefit on different hardware platforms. A more flexible means of requesting the most optimal underlying page size may evolve in future releases.

mpss.so.1(1), a preloadable shared object, can also be used to set the preferred stack and/or heap page sizes.

trapstat(1M)

```
NAME
     trapstat - report trap statistics

SYNOPSIS
     /usr/platform/  platform-name  /sbin/trapstat  [-t  |  -T  |
     -e entry]   [-C processor_set_id | -c cpulist]  [-P] [-a] [-
     r rate] [ [ interval [count]] | command | [args]]

/usr/platform/ platform-name /sbin/trapstat -l

DESCRIPTION
     The trapstat utility  gathers  and  displays  run-time  trap
     statistics  on  UltraSPARC-based systems. The default output
     is a table of trap types and CPU IDs, with each row  of  the
     table  denoting  a  trap  type  and each column of the table
     denoting a CPU. If standard output is a terminal, the  table
     contains  as many columns of data as can fit within the ter-
     minal width; if standard output is not a terminal, the table
     contains  at  most  six columns of data. By default, data is
     gathered and and displayed for all CPUs; if the data  cannot
     fit in a single table, it is printed across multiple tables.
     The set of CPUs for which data is gathered and displayed can
     be optionally specified with the -c or -C option.

     Unless the -r option or the  -a  option  is  specified,  the
     value  displayed  in  each entry of the table corresponds to
     the number of traps per second. If the -r option  is  speci-
     fied,  the value corresponds to the number of traps over the
     interval implied by the specified sampling rate; if  the  -a
     option  is  specified,  the value corresponds to the accumu-
     lated number of traps since the invocation of trapstat.

     By default, trapstat displays data once per second, and runs
     indefinitely; both of these behaviors can be optionally con-
     trolled with the  interval  and  count  parameters,  respec-
     tively.  The  interval  is  specified  in seconds; the count
     indicates the number of  intervals  to  be  executed  before
     exiting.  Alternatively,  command can be specified, in which
     case trapstat executes the provided command and continues to
     run  until the command exits.  A positive integer is assumed
     to be an interval; if the desired  command  cannot  be  dis-
     tinguished from an integer, the full path of command must be
     specified.

(continued on next page)
```

(*continued from preceding page)*

```
     UltraSPARC I, II and III handle translation lookaside buffer
     (TLB)  misses  by trapping to the operating system. TLB miss
     traps can be a significant component of overall system  per-
     formance for some workloads; the -t option provides in-depth
     information on these  traps.  When  run  with  this  option,
     trapstat  displays  both the rate of TLB miss traps  and the
     percentage of time spent processing those  traps.  Addition-
     ally,  TLB misses that hit in the translation storage buffer
     (TSB) are differentiated from TLB misses that  further  miss
     in  the  TSB.   (The  TSB  is a software structure used as a
     translation entry cache to  allow  the  TLB  to  be  quickly
     filled;  it  is  discussed  in detail in the UltraSPARC I&II
     User's Manual.) The TLB and TSB miss information is  further
     broken down into user- and kernel-mode misses.

     Workloads with working sets that exceed the  TLB  reach  may
     spend  a  significant  amount of time missing in the TLB. To
     accommodate such workloads, the  operating  system  supports
     multiple  page  sizes: larger page sizes increase the effec-
     tive TLB reach and thereby reduce the number of TLB  misses.
     To  provide  insight into the relationship between page size
     and TLB miss rate, trapstat optionally provides in-depth TLB
     miss  information  broken  down  by  page  size using the -T
     option. The information provided  by  the  -T  option  is  a
     superset  of  that provided by the -t option; only one of -t
     and -T can be specified.

OPTIONS
     The following options are supported:

     -a    Displays the number of traps as accumulating, monoton-
           ically  increasing  values  instead  of  per-second or
           per-interval rates.

     -c cpulist
           Enables trapstat only on the CPUs  specified  by  cpu-
           list.

           cpulist can be a single processor ID (for example, 4),
           a  range  of  processor  IDs  (for example, 4-6), or a
           comma separated list of processor IDs or processor  ID
           ranges (for example, 4,5,6 or 4,6-8).

     -C processor_set_id
           Enables trapstat only on the CPUs in the processor set
           specified by processor_set_id.
```

(*continued on next page)*

(*continued from preceding page)*

trapstat modifies its output to always reflect the CPUs in the specified processor set. If a CPU is added to the set, trapstat modifies its output to include the added CPU; if a CPU is removed from the set, trapstat modifies its output to exclude the removed CPU. At most one processor set can be specified.

-e entrylist

Enables trapstat only for the trap table entry or entries specified by entrylist. A trap table entry can be specified by trap number or by trap name (for example, the level-10 trap can be specified as 74, 0x4A, 0x4a, or level-10).

entrylist can be a single trap table entry or a comma separated list of trap table entries. If the specified trap table entry is not valid, trapstat prints a table of all valid trap table entries and values. A list of valid trap table entries is also found in The SPARC Architecture Manual, Version 9 and the Sun Microelectronics UltraSPARC I&II User's Manual. If the parsable option (-P) is specified in addition to the -e option, the format of the data is as follows:

Field	Contents
1	Timestamp (nanoseconds since start
2	CPU ID
3	Trap number (in hexadecimal)
4	Trap name
5	Trap rate per interval

Each field is separated with whitespace. If the format is modified, it will be modified by adding potentially new fields beginning with field 6; exant fields will remain unchanged.

-l Lists trap table entries. By default, a table is displayed containing all valid trap numbers, their names and a brief description. The trap name is used in both the default output and in the entrylist parameter for the -e argument. If the parsable option (-P) is specified in addition to the -l option, the format of the data is as follows:

Field	Contents
1	Trap number in hexadecimal
2	Trap number in decimal
3	Trap name
Remaining	Trap description

(*continued on next page)*

(*continued from preceding page)*

```
     -P    Generates parsable output. When run without other data
           gathering  modifying  options (that is, -e, -t or -T),
           trapstat's the parsable output has the following format:
           Field                          Contents
           1                              Timestamp (nanoseconds since start
           2                              CPU ID
           3                              Trap number (in hexadecimal)
           4                              Trap name
           5                              Trap rate per interval

           Each field is separated with whitespace. If the format
           is modified, it will be modified by adding potentially
           new fields beginning with field 6; extant fields  will
           remain unchanged.
     -r rate
           Explicitly sets the sampling rate to be  rate  samples
           per  second.  If  this option is specified, trapstat's
           output changes from a traps-per-second  to  traps-per-
           sampling-interval.
     -t    Enables TLB statistics.

           A table is displayed with four  principal  columns  of
           data:  itlb-miss, itsb-miss, dtlb-miss, and dtsb-miss.
           The columns contain both the rate of the corresponding
           event  and the percentage of CPU time spent processing
           the event. The percentage of CPU time is given only in
           terms   of   a   single   CPU.   The   rows   of   the   table
           correspond to CPUs, with each CPU consuming two  rows:
           one  row for user-mode events (denoted with u) and one
           row for kernel-mode events (denoted with k). For  each
           row,  the  percentage  of  CPU  time  is  totalled and
           displayed in the rightmost column. The CPUs  are  del-
           ineated with a solid line. If the parsable option (-P)
           is specified in addition to the -t option, the  format
           of the data is as follows:
           Field             Contents
           1                 Timestamp (nanoseconds since start)
           2                 CPU ID
           3                 Mode (k denotes kernel, u denotes user)
           4                 I-TLB misses
           5                 Percentage of time in I-TLB miss handler
           6                 I-TSB misses
           7                 Percentage of time in I-TSB miss handler
           8                 D-TLB misses
           9                 Percentage of time in D-TLB miss handler
           10                D-TSB misses
           11                Percentage of time in D-TSB miss handler
```

(*continued on next page)*

(*continued from preceding page)*

Each field is separated with whitespace. If the format is modified, it will be modified by adding potentially new fields beginning with field 12; extant fields will remain unchanged.

-T Enables TLB statistics, with page size information. As with the -t option, a table is displayed with four principal columns of data: itlb-miss, itsb-miss, dtlb-miss, and dtsb-miss. The columns contain both the absolute number of the corresponding event, and the percentage of CPU time spent processing the event. The percentage of CPU time is given only in terms of a single CPU. The rows of the table correspond to CPUs, with each CPU consuming two sets of rows: one set for user-level events (denoted with u) and one set for kernel-level events (denoted with k). Each set, in turn, contains as many rows as there are page sizes supported (see getpagesizes(3C)). For each row, the percentage of CPU time is totalled and displayed in the right-most column. The two sets are delineated with a dashed line; CPUs are delineated with a solid line. If the parsable option (-P) is specified in addition to the -T option, the format of the data is as follows:

Field	Contents
1	Timestamp (nanoseconds since start)
2	CPU ID
3	Mode k denotes kernel, u denotes user)
4	Page size, in decimal
5	I-TLB misses
6	Percentage of time in I-TLB miss handler
7	I-TSB misses
8	Percentage of time in I-TSB miss handler
9	D-TLB misses
10	Percentage of time in D-TLB miss handler
11	D-TSB misses
12	Percentage of time in D-TSB miss handler

Each field is separated with whitespace. If the format is modified, it will be modified by adding potentially new fields beginning with field 13; extant fields will remain unchanged.

(*continued on next page)*

(*continued from preceding page*)

EXAMPLES

Example 1: Using trapstat Without Options

When run without options, trapstat displays a table of trap types and CPUs. At most six columns can fit in the default terminal width; if (as in this example) there are more than six CPUs, multiple tables are displayed:

```
example# trapstat
vct  name                 |     cpu0     cpu1     cpu4     cpu5     cpu8     cpu9
------------------------+-------------------------------------------------------
 24 cleanwin             |     6446     4837     6368     2153     2623     1321
 41 level-1              |      100        0        0        0        1        0
 44 level-4              |        0        1        1        1        0        0
 45 level-5              |        0        0        0        0        0        0
 47 level-7              |        0        0        0        0        9        0
 49 level-9              |      100      100      100      100      100      100
 4a level-10             |      100        0        0        0        0        0
 4d level-13             |        6       10        7       16       13       11
 4e level-14             |      100        0        0        0        1        0
 60 int-vec              |     2607     2740     2642     2922     2920     3033
 64 itlb-miss            |     3129     2475     3167     1037     1200      569
 68 dtlb-miss            |   121061    86162   109838    37386    45639    20269
 6c dtlb-prot            |      997      847     1061      379      406      184
 84 spill-user-32        |     2809     2133     2739   200806   332776   454504
 88 spill-user-64        |    45819   207856    93487   228529    68373    77590
 8c spill-user-32-cln    |      784      561      767      274      353      215
 90 spill-user-64-cln    |        9       37       17       39       12       13
 98 spill-kern-64        |    62913    50145    63869    21916    28431    11738
 a4 spill-asuser-32      |     1327      947     1288      460      572      335
 a8 spill-asuser-64      |       26       48       18       54       10       14
 ac spill-asuser-32-cln  |     4580     3599     4555     1538     1978      857
 b0 spill-asuser-64-cln  |       26        0        0        2        0        0
 c4 fill-user-32         |     2862     2161     2798   191746   318115   435850
 c8 fill-user-64         |    45813   197781    89179   217668    63905    74281
 cc fill-user-32-cln     |     3802     2833     3733    10153    16419    19475
 d0 fill-user-64-cln     |      329    10105     4873    10603     4235     3649
 d8 fill-kern-64         |    62519    49943    63611    21824    28328    11693
108 syscall-32           |     2285     1634     2278      737      957      383
126 self-xcall           |      100        0        0        0        0        0
```

(*continued on next page*)

(*continued from preceding page*)

```
     vct  name                 |    cpu12    cpu13    cpu14    cpu15
     -------------------------+-----------------------------------
      24 cleanwin              |     5435     4232     6302     6104
      41 level-1               |        0        0        0        0
      44 level-4               |        2        0        0        1
      45 level-5               |        0        0        0        0
      47 level-7               |        0        0        0        0
      49 level-9               |      100      100      100      100
      4a level-10              |        0        0        0        0
      4d level-13              |       15       11       22       11
      4e level-14              |        0        0        0        0
      60 int-vec               |     2813     2833     2738     2714
      64 itlb-miss             |     2636     1925     3133     3029
      68 dtlb-miss             |    90528    70639   107786   103425
      6c dtlb-prot             |      819      675      988      954
      84 spill-user-32         |   175768    39933     2811     2742
      88 spill-user-64         |        0   241348    96907   118298
      8c spill-user-32-cln     |      681      513      753      730
      90 spill-user-64-cln     |        0       42       16       20
      98 spill-kern-64         |    52158    40914    62305    60141
      a4 spill-asuser-32       |     1113      856     1251     1208
      a8 spill-asuser-64       |        0       64       16       24
      ac spill-asuser-32-cln   |     3816     2942     4515     4381
      b0 spill-asuser-64-cln   |        0        0        0        0
      c4 fill-user-32          |   170744    38444     2876     2784
      c8 fill-user-64          |        0   230381    92941   111694
      cc fill-user-32-cln      |     8550     3790     3612     3553
      d0 fill-user-64-cln      |        0    10726     4495     5845
      d8 fill-kern-64          |    51968    40760    62053    59922
     108 syscall-32            |     1839     1495     2144     2083
     126 self-xcall            |        0        0        0        0
```

Example 2: Using trapset with CPU Filtering

The -c option can be used to limit the CPUs on which trapstat is enabled. This example limits CPU 1 and CPUs 12 through 15.

```
example# trapstat -c 1,12-15

vct  name                 |     cpu1    cpu12    cpu13    cpu14    cpu15
-------------------------+-----------------------------------------------
 24 cleanwin              |     6923     3072     2500     3518     2261
 44 level-4               |        3        0        0        1        1
 49 level-9               |      100      100      100      100      100
 4d level-13              |       23        8       14       19       14
```

(*continued on next page*)

(*continued from preceding page*)

```
      60 int-vec             |     2559     2699     2752     2688     2792
      64 itlb-miss           |     3296     1548     1174     1698     1087
      68 dtlb-miss           |   114788    54313    43040    58336    38057
      6c dtlb-prot           |     1046      549      417      545      370
      84 spill-user-32       |    66551    29480   301588    26522   213032
      88 spill-user-64       |        0   318652   111239   299829   221716
      8c spill-user-32-cln   |      856      347      331      416      293
      90 spill-user-64-cln   |        0       55       21       59       39
      98 spill-kern-64       |    66464    31803    24758    34004    22277
      a4 spill-asuser-32     |     1423      569      560      698      483
      a8 spill-asuser-64     |        0       74       32       98       46
      ac spill-asuser-32-cln |     4875     2250     1728     2384     1584
      b0 spill-asuser-64-cln |        0        2        0        1        0
      c4 fill-user-32        |    64193    28418   287516    27055   202093
      c8 fill-user-64        |        0   305016   106692   288542   210654
      cc fill-user-32-cln    |     6733     3520    15185     2396    12035
      d0 fill-user-64-cln    |        0    13226     3506    12933    11032
      d8 fill-kern-64        |    66220    31680    24674    33892    22196
     108 syscall-32          |     2446      967      817     1196      755
```

Example 3: Using trapstat with TLB Statistics

The -t option displays in-depth TLB statistics, including the amount of time spent performing TLB miss processing. The following example shows that the machine is spending 14.1 percent of its time just handling D-TLB misses:

```
example# trapstat -t
cpu m| itlb-miss %tim itsb-miss %tim | dtlb-miss %tim dtsb-miss %tim |%tim
-----+-------------------------------+-------------------------------+----
  0 u|      2571  0.3         0  0.0 |     10802  1.3         0  0.0 | 1.6
  0 k|         0  0.0         0  0.0 |    106420 13.4       184  0.1 |13.6
-----+-------------------------------+-------------------------------+----
  1 u|      3069  0.3         0  0.0 |     10983  1.2       100  0.0 | 1.6
  1 k|        27  0.0         0  0.0 |    106974 12.6        19  0.0 |12.7
-----+-------------------------------+-------------------------------+----
  2 u|      3033  0.3         0  0.0 |     11045  1.2       105  0.0 | 1.6
  2 k|        43  0.0         0  0.0 |    107842 12.7       108  0.0 |12.8
-----+-------------------------------+-------------------------------+----
  3 u|      2924  0.3         0  0.0 |     10380  1.2       121  0.0 | 1.6
  3 k|        54  0.0         0  0.0 |    102682 12.2        16  0.0 |12.2
-----+-------------------------------+-------------------------------+----
  4 u|      3064  0.3         0  0.0 |     10832  1.2       120  0.0 | 1.6
  4 k|        31  0.0         0  0.0 |    107977 13.0       236  0.1 |13.1
=====+===============================+===============================+====
 ttl |     14816  0.3         0  0.0 |    585937 14.1      1009  0.0 |14.5
```

(*continued on next page*)

(*continued from preceding page*)

Example 4: Using trapstat with TLB Statistics and Page Size Information

By specifying the -T option, trapstat shows TLB misses broken down by page size. In this example, CPU 0 is spending 7.9 percent of its time handling user-mode TLB misses on 8K pages, and another 2.3 percent of its time handling user-mode TLB misses on 64K pages.

```
example# trapstat -T -c 0
cpu m size| itlb-miss %tim itsb-miss %tim | dtlb-miss %tim dtsb-miss %tim |%tim
----------+------------------------------+------------------------------+----
  0 u   8k|      1300  0.1        15  0.0 |    104897  7.9        90  0.0 | 8.0
  0 u  64k|         0  0.0         0  0.0 |     29935  2.3         7  0.0 | 2.3
  0 u 512k|         0  0.0         0  0.0 |      3569  0.2         2  0.0 | 0.2
  0 u   4m|         0  0.0         0  0.0 |       233  0.0         2  0.0 | 0.0
- - - - - + - - - - - - - - - - - - - - - + - - - - - - - - - - - - - - - + - -
  0 k   8k|        13  0.0         0  0.0 |     71733  6.5       110  0.0 | 6.5
  0 k  64k|         0  0.0         0  0.0 |         0  0.0         0  0.0 | 0.0
  0 k 512k|         0  0.0         0  0.0 |         0  0.0       206  0.1 | 0.1
  0 k   4m|         0  0.0         0  0.0 |         0  0.0         0  0.0 | 0.0
==========+==============================+==============================+====
      ttl |      1313  0.1        15  0.0 |    210367 17.1       417  0.2 |17.5
```

Example 5: Using trapstat with Entry Filtering

By specifying the -e option, trapstat displays statistics for only specific trap types. Using this option minimizes the probe effect when seeking specific data. This example yields statistics for only the dtlb-prot and syscall-32 traps on CPUs 12 through 15:

```
example# trapstat -e dtlb-prot,syscall-32 -c 12-15
vct  name                |    cpu12    cpu13    cpu14    cpu15
-------------------------+-----------------------------------
 6c dtlb-prot            |      817      754     1018      560
108 syscall-32           |     1426     1647     2186     1142

vct  name                |    cpu12    cpu13    cpu14    cpu15
-------------------------+-----------------------------------
 6c dtlb-prot            |     1085      996      800      707
108 syscall-32           |     2578     2167     1638     1452
```

(*continued on next page*)

(*continued from preceding page)*

Example 6: Using trapstat with a Higher Sampling Rate

The following example uses the -r option to specify a sampling rate of 1000 samples per second, and filter only for the level-10 trap. Additionally, specifying the -P option yields parsable output.

Notice the timestamp difference between the level-10 events: 9,998,000 nanoseconds and 10,007,000 nanoseconds. These level-10 events correspond to the system clock, which by default ticks at 100 hertz (that is, every 10,000,000 nanoseconds).

```
example# trapstat -e level-10 -P -r 1000
1070400 0 4a level-10 0
2048600 0 4a level-10 0
3030400 0 4a level-10 1
4035800 0 4a level-10 0
5027200 0 4a level-10 0
6027200 0 4a level-10 0
7027400 0 4a level-10 0
8028200 0 4a level-10 0
9026400 0 4a level-10 0
10029600 0 4a level-10 0
11028600 0 4a level-10 0
12024000 0 4a level-10 0
13028400 0 4a level-10 1
14031200 0 4a level-10 0
15027200 0 4a level-10 0
16027600 0 4a level-10 0
17025000 0 4a level-10 0
18026000 0 4a level-10 0
19027800 0 4a level-10 0
20025600 0 4a level-10 0
21025200 0 4a level-10 0
22025000 0 4a level-10 0
23035400 0 4a level-10 1
24027400 0 4a level-10 0
25026000 0 4a level-10 0
26027000 0 4a level-10 0
```

(*continued on next page)*

(*continued from preceding page*)

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWcsu
Interface Stability	
Human Readable Output	Unstable
Parsable Output	Evolving

SEE ALSO

lockstat(1M), pmap(1), psrset(1M), psrinfo(1M), pbind(1M), ppgsz(1), getpagesizes(3C)

Sun Microelectronics UltraSPARC I&II User's Manual, January 1997, STP1031,

The SPARC Architecture Manual, Version 9, 1994, Prentice-Hall.

NOTES

When enabled, trapstat induces a varying probe effect, depending on the type of information collected. While the precise probe effect depends upon the specifics of the hardware, the following table can be used as a rough guide:

Option	Approximate probe effect
default	3-5% per trap
-e	3-5% per specified trap
-t, -T	40-45% per TLB miss trap hitting in the TSB, 25-30% per TLB miss trap missing in the TSB

These probe effects are per trap not for the system as a whole. For example, running trapstat with the default options on a system that spends 7% of total time handling traps induces a performance degradation of less than one half of one percent; running trapstat with the -t or -T option on a system spending 5% of total time processing TLB misses induce a performance degradation of no more than 2.5%.

(*continued on next page*)

```
(continued from preceding page)

     When run with the -t or -T option, trapstat accounts for its
     probe  effect when calculating the %tim fields. This assures
     that the %tim fields are a reasonably accurate indicator  of
     the  time a given workload is spending handling TLB misses -
     regardless of the perturbing presence of trapstat.

     While the %tim fields include the explicit cost of executing
     the TLB miss handler, they do not include the implicit costs
     of TLB miss traps (for example, pipeline effects, cache pol-
     lution,  etc).  These implicit costs become more significant
     as the trap rate grows; if high  %tim  values  are  reported
     (greater  than  50%),  you can accurately infer that much of
     the balance of time is being spent on the implicit costs  of
     the TLB miss traps.

     Due to the potential system wide degradation  induced,  only
     the super-user can run trapstat.

     Due to the limitation of the underlying statistics gathering
     methodology,  only  one  instance  of  trapstat can run at a
     time.
```

mpss.so.1(1)

```
NAME
     mpss.so.1 - shared object for setting preferred page size

SYNOPSIS
     mpss.so.1

DESCRIPTION
     The mpss.so.1 shared object provides a means  by  which  the
     preferred  stack  and/or  heap  page size can be selectively
     configured for launched processes and their descendants.  To
     enable  mpss.so.1,  the following string needs to be present
     in the environment (see ld.so.1(1)) along with one  or  more
     MPSS (Multiple Page Size Support) environment variables:

(continued on next page)
```

(*continued from preceding page)*

```
     LD_PRELOAD=$LD_PRELOAD:mpss.so.1

ENVIRONMENT VARIABLES
     Once preloaded, the mpss.so.1 shared object reads  the  fol-
     lowing environment variables to determine any preferred page
     size requirements and any processes these  may  be  specific
     to.

     MPSSHEAP=size

     MPSSSTACK=size
           MPSSHEAP and  MPSSSTACK  specify  the  preferred  page
           sizes for the heap and stack, respectively. The speci-
           fied  page  size(s)  are  applied   to   all   created
           processes.

           size must be a supported page size  (see  pagesize(1))
           or  0,  in  which  case  the  system  will  select  an
           appropriate page size (see memcntl(2)).

           size can be qualified with K, M, G, or  T  to  specify
           Kilobytes,  Megabytes, Gigabytes, or Terabytes respec-
           tively.

     MPSSCFGFILE=config-file
           config-file is a text file which contains one or  more
           mpss configuration entries of the form:


           exec-name:heap-size:stack-size

           exec-name specifies the name of an application or exe-
           cutable.  The corresponding preferred page size(s) are
           set for newly created processes (see  getexecname(3C))
           that match the first exec-name found in the file.

           exec-name can be a full pathname, a  base  name  or  a
           pattern  string. See File Name Generation in sh(1) for
           a discussion of pattern matching.

           If heap-size and/or stack-size are not specified,  the
           corresponding preferred page size(s) will not be set.

           MPSSCFGFILE  takes  precedence   over   MPSSHEAP   and
           MPSSSTACK.
```

(*continued on next page)*

(*continued from preceding page)*

```
     MPSSERRFILE=pathname
           By default, error messages are logged  via  syslog(3C)
           using   level   LOG_ERR   and  facility  LOG_USER.  If
           MPSSERRFILE  contains  a  valid  pathname   (such   as
           /dev/stderr),  error  messages  will  be  logged there
           instead.

EXAMPLES
     Example  1:   Configuring   preferred   page   sizes   using
     MPSSCFGFILE

     The following Bourne shell commands  (see  sh(1))  configure
     the  preferred  page  sizes  to a select set of applications
     with exec names that begin with foo, using  the  MPSSCFGFILE
     environment  variable. The MPSS configuration file, mpsscfg,
     is assumed to have been previously created via a text editor
     like  vi(1). The cat(1) command is only dumping out the con-
     tents.

     example$ LD_PRELOAD=$LD_PRELOAD:mpss.so.1
     example$ MPSSCFGFILE=mpsscfg
     example$ export LD_PRELOAD MPSSCFGFILE
     example$ cat $MPSSCFGFILE
     foo*:512K:64K

     Once the application has been started,  pmap  (see  proc(1))
     can be used to view the actual page sizes configured:

     example$ foobar &
     example$ pmap -s ‘pgrep foobar‘

     If the desired page size is not  configured  (shown  in  the
     pmap output), it may be due to errors in the MPSS configura-
     tion file or environment variables. Check the error log  (by
     default: /var/adm/messages) for errors.

     If no errors can be found, resource or alignment constraints
     may be responsible. See the NOTES section.
```

(*continued on next page)*

(*continued from preceding page)*

Example 2: Configuring preferred page sizes using MPSSHEAP and MPSSSTACK

The following Bourne shell commands configure 512K heap and 64K stack preferred page sizes for all applications using the MPSSHEAP and MPSSSTACK environment variables.

```
example$ LD_PRELOAD=$LD_PRELOAD:mpss.so.1
example$ MPSSHEAP=512K
example$ MPSSSTACK=64K
example$ export LD_PRELOAD MPSSHEAP MPSSSTACK
```

Example 3: Precedence rules (continuation from Example 2)

The preferred page size configuration in MPSSCFGFILE overrides MPSSHEAP and MPSSTACK. Appending the following commands to those in Example 2 would mean that all applications will be configured with 512K heap and 64K stack preferred page sizes with the exception of those applications, the ls command, and all applications beginning with ora, in the configuration file.

```
example$ MPSSCFGFILE=mpsscfg2
example$ export MPSSCFGFILE
example$ cat $MPSSCFGFILE
ls::
ora*:4m:4m
```

FILES

/usr/lib/ld/map.bssalign
: A template link-editor mapfile for aligning bss (see NOTES).

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUNWesu (32-bit)
	SUNWesxu (64-bit)
Interface Stability	Evolving

(*continued on next page)*

```
(continued from preceding page)

SEE ALSO
     cat(1), ld(1), ld.so.1(1), pagesize(1),  ppgsz(1),  proc(1),
     sh(1), vi(1), exec(2), fork(2), memcntl(2), getexecname(3C),
     getpagesize(3C), syslog(3C), proc(4), attributes(5)

NOTES
     The heap and stack preferred page  sizes  are  inherited.  A
     child  process  has  the  same  preferred  page sizes as its
     parent. On exec(2), the preferred page sizes are set back to
     the  default  system  page size unless a preferred page size
     has been configured via the mpss shared object.

     ppgsz(1), a proc tool, can also be used to set the preferred
     stack  and/or heap page sizes. It cannot selectively config-
     ure the page size for descendents based on name matches.

     See also NOTES under ppgsz(1).
```

memcntl(2)

```
NAME
     memcntl - memory management control

SYNOPSIS
     #include <sys/types.h>
     #include <sys/mman.h>

     int memcntl(caddr_t addr, size_t len, int cmd, caddr_t  arg,
     int attr, int mask);

DESCRIPTION
     The memcntl() function allows the calling process to apply a
     variety of control operations over the address space identi-
     fied by the  mappings  established  for  the  address  range
     [addr, addr + len).

(continued on next page)
```

(*continued from preceding page)*

The addr argument must be a multiple of the pagesize as returned by sysconf(3C). The scope of the control operations can be further defined with additional selection criteria (in the form of attributes) according to the bit pattern contained in attr.

The following attributes specify page mapping selection criteria:

SHARED
Page is mapped shared.

PRIVATE
Page is mapped private.

The following attributes specify page protection selection criteria. The selection criteria are constructed by a bit-wise OR operation on the attribute bits and must match exactly.

PROT_READ
Page can be read.

PROT_WRITE
Page can be written.

PROT_EXEC
Page can be executed.

The following criteria may also be specified:

PROC_TEXT
Process text.

PROC_DATA
Process data.

The PROC_TEXT attribute specifies all privately mapped segments with read and execute permission, and the PROC_DATA attribute specifies all privately mapped segments with write permission.

(*continued on next page)*

(*continued from preceding page)*

Selection criteria can be used to describe various abstract memory objects within the address space on which to operate. If an operation shall not be constrained by the selection criteria, attr must have the value 0.

The operation to be performed is identified by the argument cmd. The symbolic names for the operations are defined in <sys/mman.h> as follows:

MC_LOCK

Lock in memory all pages in the range with attributes attr. A given page may be locked multiple times through different mappings; however, within a given mapping, page locks do not nest. Multiple lock operations on the same address in the same process will all be removed with a single unlock operation. A page locked in one process and mapped in another (or visible through a different mapping in the locking process) is locked in memory as long as the locking process does neither an implicit nor explicit unlock operation. If a locked mapping is removed, or a page is deleted through file removal or truncation, an unlock operation is implicitly performed. If a writable MAP_PRIVATE page in the address range is changed, the lock will be transferred to the private page.

The arg argument is not used, but must be 0 to ensure compatibility with potential future enhancements.

MC_LOCKAS

Lock in memory all pages mapped by the address space with attributes attr. The addr and len arguments are not used, but must be NULL and 0 respectively, to ensure compatibility with potential future enhancements. The arg argument is a bit pattern built from the flags:

The value of arg determines whether the pages to be locked are those currently mapped by the address space, those that will be mapped in the future, or both. If MCL_FUTURE is specified, then all mappings subsequently added to the address space will be locked, provided sufficient memory is available.

(*continued on next page)*

(*continued from preceding page)*

MCL_CURRENT
Lock current mappings.

MCL_FUTURE
Lock future mappings.

MC_SYNC
Write to their backing storage locations all modified pages in the range with attributes attr. Optionally, invalidate cache copies. The backing storage for a modified MAP_SHARED mapping is the file the page is mapped to; the backing storage for a modified MAP_PRIVATE mapping is its swap area. The arg argument is a bit pattern built from the flags used to control the behavior of the operation:

MS_ASYNC
Perform asynchronous writes.

MS_SYNC
Perform synchronous writes.

MS_INVALIDATE
Invalidate mappings.

MS_ASYNC Return immediately once all write operations are scheduled; with MS_SYNC the function will not return until all write operations are completed.

MS_INVALIDATE Invalidate all cached copies of data in memory, so that further references to the pages will be obtained by the system from their backing storage locations. This operation should be used by applications that require a memory object to be in a known state.

MC_UNLOCK
Unlock all pages in the range with attributes attr. The arg argument is not used, but must be 0 to ensure compatibility with potential future enhancements.

MC_UNLOCKAS
Remove address space memory locks and locks on all pages in the address space with attributes attr. The addr, len, and arg arguments are not used, but must be NULL, 0 and 0, respectively, to ensure compatibility with potential future enhancements.

(*continued on next page)*

```
(continued from preceding page)

     MC_HAT_ADVISE
           Advise system how a region of user-mapped memory  will
           be  accessed.  The  arg  argument  is interpreted as a
           "struct memcntl_mha  *".  The  following  members  are
           defined in a struct memcntl_mha:

System Calls                                            memcntl(2)
           uint_t mha_cmd;
           uint_t mha_flags;
           size_t mha_pagesize;

           The accepted values for mha_cmd are:
           MHA_MAPSIZE_VA
           MHA_MAPSIZE_STACK
           MHA_MAPSIZE_BSSBRK

           The mha_flags member is reserved for  future  use  and
           must  always be set to 0. The mha_pagesize member must
           be a valid size as obtained from  getpagesizes(3C)  or
           the  constant value 0 to allow the system to choose an
           appropriate hardware address translation mapping size.

           MHA_MAPSIZE_VA sets  the  preferred  hardware  address
           translation  mapping size of the region of memory from
           addr to addr + len. Both addr and len must be  aligned
           to   an   mha_pagesize   boundary.   The   entire  virtual
           address region from addr to addr + len must  not  have
           any   holes.   Permissions   within   each   mha_pagesize-
           aligned portion of  the  region  must  be  consistent.
           When  a  size of 0 is specified, the system selects an
           appropriate size based on the size  and  alignment  of
           the  memory  region, type of processor, and other con-
           siderations.

           MHA_MAPSIZE_STACK sets the preferred hardware  address
           translation  mapping  size  of the process main thread
           stack segment. The addr and len arguments must be NULL
           and 0, respectively.

           MHA_MAPSIZE_BSSBRK sets the preferred hardware address
           translation mapping size of the process heap. The addr
           and len arguments must be NULL  and  0,  respectively.
           See  the NOTES section of the ppgsz(1) manual page for
           additional information on process heap alignment.

           The attr argument must  be  0  for  all  MC_HAT_ADVISE
           operations.
(continued on next page)
```

(*continued from preceding page)*

The mask argument must be 0; it is reserved for future use.

Locks established with the lock operations are not inherited by a child process after fork(2). The memcntl() function fails if it attempts to lock more memory than a system-specific limit.

Due to the potential impact on system resources, all operations except MC_SYNC are restricted to processes with superuser effective user ID.

USAGE

The memcntl() function subsumes the operations of plock(3C) and mctl(3UCB).

MC_HAT_ADVISE is intended to improve performance of applications that use large amounts of memory on processors that support multiple hardware address translation mapping sizes; however, it should be used with care. Not all processors support all sizes with equal efficiency. Use of larger sizes may also introduce extra overhead that could reduce performance or available memory. Using large sizes for one application may reduce available resources for other applications and result in slower system wide performance.

RETURN VALUES

Upon successful completion, memcntl() returns 0; otherwise, it returns -1 and sets errno to indicate an error.

ERRORS

The memcntl() function will fail if:

EAGAIN
When the selection criteria match, some or all of the memory identified by the operation could not be locked when MC_LOCK or MC_LOCKAS was specified, some or all mappings in the address range [addr, addr + len) are locked for I/O when MC_HAT_ADVISE was specified, or the system has insufficient resources when MC_HAT_ADVISE was specified.

EBUSY When the selection criteria match, some or all of the addresses in the range [addr, addr + len) are locked and MC_SYNC with the MS_INVALIDATE option was specified.

(*continued on next page)*

(*continued from preceding page)*

EINVAL
The addr argument specifies invalid selection criteria or is not a multiple of the page size as returned by sysconf(3C); the addr and/or len argument does not have the value 0 when MC_LOCKAS or MC_UNLOCKAS is specified; the arg argument is not valid for the function specified; mha_pagesize or mha_cmd is invalid; or MC_HAT_ADVISE is specified and not all pages in the specified region have the same access permissions within the given size boundaries.

ENOMEM
When the selection criteria match, some or all of the addresses in the range [addr, addr + len) are invalid for the address space of a process or specify one or more pages which are not mapped.

EPERM The process's effective user ID is not superuser and MC_LOCK, MC_LOCKAS, MC_UNLOCK, or MC_UNLOCKAS was specified.

ATTRIBUTES
See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
MT-Level	MT-Safe

SEE ALSO
ppgsz(1), fork(2) mmap(2), mprotect(2), getpagesizes(3C), mctl(3UCB), mlock(3C), mlockall(3C), msync(3C), plock(3C), sysconf(3C), attributes(5)

About the Author

Richard has over 15 years of UNIX experience including application design, kernel development, and performance analysis. Richard specializes in operating system tools and architecture.

Ordering Sun Documents

The SunDocs℠ program provides more than 250 manuals from Sun Microsystems, Inc. If you live in the United States, Canada, Europe, or Japan, you can purchase documentation sets or individual manuals through this program.

Accessing Sun Documentation Online

The `docs.sun.com` web site enables you to access Sun technical documentation online. You can browse the `docs.sun.com` archive or search for a specific book title or subject. The URL is `http://docs.sun.com/`

To reference Sun BluePrints OnLine articles, visit the Sun BluePrints OnLine Web site at: `http://www.sun.com/blueprints/online.html`