```
************************************************************
    5814 Fri Feb  3 18:30:56 2017
new/usr/src/man/man9/vmem.9
code review from Josh and Robert
************************************************************
.\"
.\" CDDL HEADER START
.\"
.\" The contents of this file are subject to the terms of the
.\" Common Development and Distribution License (the "License").
.\" You may not use this file except in compliance with the License.
.\"
.\" You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
.\" or http://www.opensolaris.org/os/licensing.
.\" See the License for the specific language governing permissions
.\" and limitations under the License.
.\"
.\" When distributing Covered Code, include this CDDL HEADER in each
.\" file and include the License file at usr/src/OPENSOLARIS.LICENSE.
.\" If applicable, add the following below this CDDL HEADER, with the
.\" fields enclosed by brackets "[]" replaced with your own identifying
.\" information: Portions Copyright [yyyy] [name of copyright owner]
.\"
.\" CDDL HEADER END
.\"
.\"
.\" Copyright 2010 Sun Microsystems, Inc.  All rights reserved.
.\" Use is subject to license terms.
.\"
.\" Copyright (c) 2012, 2015 by Delphix. All rights reserved.
.\" Copyright (c) 2012, Joyent, Inc. All rights reserved.
.\"
.\" The text of this is derived from section 1 of the big theory statement in
.\" usr/src/uts/common/os/vmem.c, the traditional location of this text.  They
.\" should largely be updated in tandem.
.Dd Jan 18, 2017
.Dt VMEM 9
.Os
.Sh NAME
.Nm vmem
.Nd virtual memory allocator
.Sh DESCRIPTION
.Ss Overview
An address space is divided into a number of logically distinct pieces, or
.Em arenas :
text, data, heap, stack, and so on.
Within these
arenas we often subdivide further; for example, we use heap addresses
not only for the kernel heap
.Po
.Fn kmem_alloc
space
.Pc ,
but also for DVMA,
.Fn bp_mapin ,
.Pa /dev/kmem ,
and even some device mappings.
.Pp
The kernel address space, therefore, is most accurately described as
a tree of arenas in which each node of the tree
.Em imports
some subset of its parent.
The virtual memory allocator manages these arenas
and supports their natural hierarchical structure.
.Ss Arenas
An arena is nothing more than a set of integers.  These integers most
```

```
commonly represent virtual addresses, but in fact they can represent
anything at all.  For example, we could use an arena containing the
integers minpid through maxpid to allocate process IDs.  For uses of this
nature, prefer
.Xr id_space 9F
instead.
.Pp
.Fn vmem_create
and
.Fn vmem_destroy
create and destroy vmem arenas.  In order to differentiate between arenas used
for addresses and arenas used for identifiers, the
for adresses and arenas used for identifiers, the
.Dv VMC_IDENTIFIER
flag is passed to
.Fn vmem_create .
This prevents identifier exhaustion from being diagnosed as general memory
failure.
.Ss Spans
We represent the integers in an arena as a collection of
.Em spans ,
or contiguous ranges of integers.  For example, the kernel heap consists of
just one span:
.Li "[kernelheap, ekernelheap)" .
Spans can be added to an arena in two ways: explicitly, by
.Fn vmem_add ;
.Fn vmem_add ,
or implicitly, by importing, as described in
.Sx Imported Memory
below.
.Ss Segments
Spans are subdivided into
.Em segments ,
each of which is either allocated or free.  A segment, like a span, is a
contiguous range of integers.  Each allocated segment
.Li "[addr, addr + size)"
represents exactly one
.Li "vmem_alloc(size)"
that returned
.Sy addr .
Free segments represent the space between allocated segments.  If two free
segments are adjacent, we coalesce them into one larger segment; that is, if
segments
.Li "[a, b)"
and
.Li "[b, c)"
are both free, we merge them into a single segment
.Li "[a, c)" .
The segments within a span are linked together in increasing\-address
order so we can easily determine whether coalescing is possible.
.Pp
Segments never cross span boundaries.  When all segments within an imported
span become free, we return the span to its source.
.Ss Imported Memory
As mentioned in the overview, some arenas are logical subsets of
other arenas.  For example,
.Sy kmem_va_arena
(a virtual address cache
that satisfies most
.Fn kmem_slab_create
requests) is just a subset of
.Sy heap_arena
(the kernel heap) that provides caching for the most common slab sizes.  When
.Sy kmem_va_arena
runs out of virtual memory, it
.Em imports
```

```
more from the heap; we say that
.Em imports more from the heap; we
say that
.Sy heap_arena
is the
.Em "vmem source"
for
.Em "vmem source" for
.Sy kmem_va_arena.
.Fn vmem_create
allows you to specify any existing vmem arena as the source for your new
arena.  Topologically, since every arena is a child of at most one source, the
set of all arenas forms a collection of trees.
.Ss Constrained Allocations
Some vmem clients are quite picky about the kind of address they want.
For example, the DVMA code may need an address that is at a particular
phase with respect to some alignment (to get good cache coloring), or
that lies within certain limits (the addressable range of a device),
or that doesn’t cross some boundary (a DMA counter restriction) \(em
or all of the above.
.Fn vmem_xalloc
allows the client to specify any or all of these constraints.
.Ss The Vmem Quantum
Every arena has a notion of
.Sq quantum ,
specified at
.Fn vmem_create
time, that defines the arena’s minimum unit of currency.  Most commonly the
quantum is either 1 or
.Dv PAGESIZE ,
but any power of 2 is legal.  All vmem allocations are guaranteed to be
quantum\-aligned.
.Ss Relationship to the Kernel Memory Allocator
Every kmem cache has a vmem arena as its slab supplier.  The kernel memory
allocator uses
.Fn vmem_alloc
and
.Fn vmem_free
to create and destroy slabs.
.Sh SEE ALSO
.Xr id_space 9F ,
.Xr vmem_add 9F ,
.Xr vmem_alloc 9F ,
.Xr vmem_contains 9F ,
.Xr vmem_create 9F ,
.Xr vmem_walk 9F
.Pp
.Rs
.%A Jeff Bonwick
.%A Jonathan Adams
.%T Magazines and vmem: Extending the Slab Allocator to Many CPUs and Arbitrary
.%J Proceedings of the 2001 Usenix Conference
.%U http://www.usenix.org/event/usenix01/bonwick.html
.Re
```

```
**********************************************************
    1572 Fri Feb  3 18:30:57 2017
new/usr/src/man/man9f/vmem_add.9f
code review from Josh and Robert
**********************************************************
.\"
.\" This file and its contents are supplied under the terms of the
.\" Common Development and Distribution License ("CDDL"), version 1.0.
.\" You may only use this file in accordance with the terms of version
.\" 1.0 of the CDDL.
.\"
.\" A full copy of the text of the CDDL should have accompanied this
.\" source.  A copy of the CDDL is also available via the Internet at
.\" http://www.illumos.org/license/CDDL.
.\"
.\"
.\" Copyright 2017, Richard Lowe.
.\"
.Dd Jan 18, 2017
.Dt VMEM_ADD 9F
.Os
.Sh NAME
.Nm vmem_add
.Nd add spans to a vmem arena
.Sh SYNOPSIS
.In sys/vmem.h
.Ft void *
.Fo vmem_add
.Fa "vmem_t *vmp"
.Fa "void *vaddr"
.Fa "size_t size"
.Fa "int vmflag"
.Fc
.Sh INTERFACE LEVEL
illumos DDI specific
.Sh PARAMETERS
.Bl -tag -width Ds
.It Fa vmp
The vmem arena to which the span should be added.
.It Fa vaddr
The base address of the span to add.
.It Fa size
The size of the span to add.
The size of the span to add
.It Fa vmflag
Flags affecting the allocation of the span to add.
.El
.Sh DESCRIPTION
The
.Fn vmem_add
function adds
.Fa size
bytes starting at
.Fa vaddr
to a vmem arena from which future calls to
.Fn vmem_alloc
may allocate.
.Pp
.Dv VM_SLEEP
or
.Dv VM_NOSLEEP
must be specified in
.Fa vmflag ,
and indicate whether the addition may block.
must be specified, and indicate whether the addition may block.
.Sh CONTEXT
```

```
This function can be called from either user or kernel context.
If the
.Dv VM_NOSLEEP
flag is specified, it may also be called from interrupt context.
.Sh RETURN VALUES
Upon success
.Fn vmem_add
returns
.Fa vaddr .
On failure,
.Dv NULL
is returned.
.Sh SEE ALSO
.Xr vmem 9 ,
.Xr vmem_alloc 9F ,
.Xr vmem_create 9F
```

```
**********************************************************
    4086 Fri Feb  3 18:30:59 2017
new/usr/src/man/man9f/vmem_alloc.9f
code review from Josh and Robert
**********************************************************
.\"
.\" This file and its contents are supplied under the terms of the
.\" Common Development and Distribution License ("CDDL"), version 1.0.
.\" You may only use this file in accordance with the terms of version
.\" 1.0 of the CDDL.
.\"
.\" A full copy of the text of the CDDL should have accompanied this
.\" source.  A copy of the CDDL is also available via the Internet at
.\" http://www.illumos.org/license/CDDL.
.\"
.\"
.\" Copyright 2017, Richard Lowe.
.\"
.Dd Jan 18, 2017
.Dt VMEM_ALLOC 9F
.Os
.Sh NAME
.Nm vmem_alloc ,
.Nm vmem_xalloc ,
.Nm vmem_free ,
.Nm vmem_xfree
.Nd allocate and free segments from a vmem arena
.Sh SYNOPSIS
.In sys/vmem.h
.Ft void *
.Fo vmem_alloc
.Fa "vmem_t *vmp"
.Fa "size_t size"
.Fa "int vmflag"
.Fc
.Ft void *
.Fo vmem_xalloc
.Fa "vmem_t *vmp"
.Fa "size_t size"
.Fa "size_t align_arg"
.Fa "size_t phase"
.Fa "size_t nocross"
.Fa "void *minaddr"
.Fa "void *maxaddr"
.Fa "int vmflag"
.Fc
.Ft void
.Fo vmem_free
.Fa "vmem_t *vmp"
.Fa "void *vaddr"
.Fa "size_t size"
.Fc
.Ft void
.Fo vmem_xfree
.Fa "vmem_t *vmp"
.Fa "void *vaddr"
.Fa "size_t size"
.Fc
.Sh INTERFACE LEVEL
illumos DDI specific
.Sh PARAMETERS
.Bl -tag -width Ds
.It Fa vmp
The vmem arena from which to allocate or free.
.It Fa size
The size of the segment to allocate or free.
```

```
.It Fa vmflag
A bitmask of flags controlling the behaviour of the allocation.
There are two meaningful groups of flags.
.Dv VM_SLEEP
or
.Dv VM_NOSLEEP
must be specified, and indicate whether the allocation may block.  A
.Dv VM_SLEEP
allocation can never fail but may block indefinitely.
.Pp
The allocation policy may be specified by one of the following flags:
.Bl -tag -width Ds
.It Dv VM_BESTFIT
Take the segment from the smallest free segment that could satisfy this allocati
Take the segment from the smallest free segment that could satisfy this allocati
.It Dv VM_FIRSTFIT
Take the segment from the first free segment found that could satisfy this
allocation.
.It Dv VM_NEXTFIT
Take the segment from the segment after the one previously allocated.  This
provides sequential behaviour useful when allocating identifiers from a
.Dv VMC_IDENTIFIER
arena.
.It Dv VM_ENDALLOC
May be specified in combination with
.Dv VM_BESTFIT ,
.Dv VM_FIRSTFIT
or the default policy to indicate that the higher addresses should be
preferred.
.El
.Pp
The default (unnamed) allocation policy is
.Dq "instant fit" ,
The default (un\-named) allocation policy is
.Dq instant fit
an approximation of
.Dv VM_BESTFIT
in guaranteed constant time.
.It Fa align_arg
The minimum alignment of the allocation.  If 0,
The minimum alignment of the allocation.  If
.Ql 0
the allocated segment will be aligned as the arena's quantum.
.It Fa phase
The allocated segment must be
.Fa phase
bytes from the alignment boundary.
.It Fa nocross
The allocated segment may not straddle a
.Fa nocross
alignment boundary.
.It Fa minaddr
The minimum address at which the segment may be allocated.
.It Fa maxaddr
The maximum address which may be included in the segment.
.It Fa vaddr
The address of the segment which
.Fn vmem_free
or
.Fn vmem_xfree
should free.
.El
.Sh DESCRIPTION
The
.Fn vmem_alloc
and
```

```
.Fn vmem_xalloc
functions allocate a segment of
.Fa size
length from the vmem arena
.Fa vmp .
.Pp
The
.Fa vmflag
argument controls the behaviour of the allocation, as described in
.Sx PARAMETERS .
argument controls the behaviour of the allocation.  As described in
.Sx PARAMETERS
.Pp
For allocations with complex requirements, such as those used for DMA
.Fn vmem_xalloc
takes additional arguments allowing those requirements to be expressed.
.Pp
Segments allocated with
.Fn vmem_xalloc
must always be freed with
.Fn vmem_xfree ,
since these allocations are uncached.
.Sh CONTEXT
This function can be called from either user or kernel context.
If the
.Dv VM_NOSLEEP
flag is specified, it may also be called from interrupt context.
.Sh RETURN VALUES
Upon successful completion the
.Fn vmem_alloc
and
.Fn vmem_xalloc
functions return a pointer to the beginning of the allocated segment.  In the
case of a
.Dv VMC_IDENTIFIER
arena, the address of this pointer is the meaningful component, not the value
to which it points.
.Pp
On failure,
.Dv NULL
is returned.
When the
.Dv VM_SLEEP
flag is specified, these functions can never fail (but may block forever).
.Sh SEE ALSO
.Xr vmem 9 ,
.Xr vmem_contains 9F ,
#endif /* ! codereview */
.Xr vmem_create 9F
```

```
**********************************************************
    1264 Fri Feb  3 18:31:01 2017
new/usr/src/man/man9f/vmem_contains.9f
code review from Josh and Robert
**********************************************************
.\"
.\" This file and its contents are supplied under the terms of the
.\" Common Development and Distribution License ("CDDL"), version 1.0.
.\" You may only use this file in accordance with the terms of version
.\" 1.0 of the CDDL.
.\"
.\" A full copy of the text of the CDDL should have accompanied this
.\" source.  A copy of the CDDL is also available via the Internet at
.\" http://www.illumos.org/license/CDDL.
.\"
.\"
.\" Copyright 2017, Richard Lowe.
.\"
.Dd Jan 18, 2017
.Dt VMEM_CONTAINS 9F
.Os
.Sh NAME
.Nm vmem_contains
.Nm vmem_contains ,
.Nd check for membership in a vmem arena
.Sh SYNOPSIS
.In sys/vmem.h
.Ft int
.Fo vmem_contains
.Fa "vmem_t *vmp"
.Fa "void *vaddr"
.Fa "size_t *size"
.Fc
.Sh INTERFACE LEVEL
illumos DDI specific
.Sh PARAMETERS
.Bl -tag -width Ds
.It Fa vmp
The vmem arena
.It Fa vaddr
address of the segment to query.
.It Fa size
size of the segment to query
.El
.Sh DESCRIPTION
The
.Fn vmem_contains
function checks whether a segment of
.Fa size
bytes at
.Fa vaddr
exists within the vmem arena
.Fa vmp .
.Sh CONTEXT
This function may be called from user or kernel context.
.Sh RETURN VALUES
.Fn vmem_contains
returns non\-0 if the segment exists, and 0 otherwise.
.Sh SEE ALSO
.Xr vmem 9 ,
.Xr vmem_add 9F ,
.Xr vmem_alloc 9F ,
.Xr vmem_create 9F
```

```
**********************************************************
    4846 Fri Feb  3 18:31:02 2017
new/usr/src/man/man9f/vmem_create.9f
code review from Josh and Robert
**********************************************************
.\"
.\" This file and its contents are supplied under the terms of the
.\" Common Development and Distribution License ("CDDL"), version 1.0.
.\" You may only use this file in accordance with the terms of version
.\" 1.0 of the CDDL.
.\"
.\" A full copy of the text of the CDDL should have accompanied this
.\" source.  A copy of the CDDL is also available via the Internet at
.\" http://www.illumos.org/license/CDDL.
.\"
.\"
.\" Copyright 2017, Richard Lowe.
.\"
.Dd Jan 18, 2017
.Dt VMEM_CREATE 9F
.Os
.Sh NAME
.Nm vmem_create ,
.Nm vmem_xcreate ,
.Nm vmem_destroy
.Nd create and destroy vmem arenas
.Sh SYNOPSIS
.In sys/vmem.h
.Vt "typedef struct vmem vmem_t;"
.Vt "typedef void *(vmem_alloc_t)(vmem_t *, size_t, int);"
.Vt "typedef void (vmem_free_t)(vmem_t *, void *, size_t);"
.Vt "typedef void *(vmem_ximport_t)(vmem_t *, size_t *, size_t, int);"
.Ft vmem_t *
.Fo vmem_create
.Fa "const char *name"
.Fa "void *base"
.Fa "size_t size"
.Fa "size_t quantum"
.Fa "vmem_alloc_t *afunc"
.Fa "vmem_free_t *ffunc"
.Fa "vmem_t *source"
.Fa "size_t qcache_max"
.Fa "int vmflag"
.Fc
.Ft vmem_t *
.Fo vmem_xcreate
.Fa "const char *name"
.Fa "void *base"
.Fa "size_t size"
.Fa "size_t quantum"
.Fa "vmem_ximport_t *afunc"
.Fa "vmem_free_t *ffunc"
.Fa "vmem_t *source"
.Fa "size_t qcache_max"
.Fa "int vmflag"
.Fc
.Ft void
.Fo vmem_destroy
.Fa "vmem_t *vmp"
.Fc
.Sh INTERFACE LEVEL
illumos DDI specific
.Sh PARAMETERS
.Bl -tag -width Ds
.It Fa name
A character string giving a name to the vmem
```

```
arena to be created.
.It Fa base
An address indicating the lowest possible value in the arena.
.It Fa size
The size of the arena to create.
The size of the arena to create
.It Fa quantum
The arena’s
.Dq quantum .
The granularity of the arena.  The amount allocated at minimum by each
request.  Must be a power of 2.
The granularity of the arena.  The amount allocated at minimum by each request.
.It Fa afunc
A function which is called to import new spans from
.Fa source ,
which may be
.Fa source .
Which may be
.Dv NULL
if this arena does not import from another.
When calling
.Fn vmem_create
.Fa afunc
is an
.Vt vmem_alloc_t
a function taking three parameters and returning a pointer to
.Vt void
(the imported space):
.Bl -tag -width Ds
.It Fa "vmem_t *"
The source arena from which we’ll import.  The
.Fa source
argument to
.Fn vmem_create .
.It Fa size_t
The size to import
.It Fa int
The
.Fa vmflag
argument used for the import.
.El
.Pp
When calling
.Fn vmem_xcreate
.Fa afunc
is an
.Vt vmem_ximport_t
a function taking four parameters and returning a pointer to
.Vt void
(the imported space):
.Bl -tag -width Ds
.It Fa "vmem_t *"
The source arena from which we’ll import.  The
.Fa source
argument to
.Fn vmem_xcreate .
.It Fa "size_t *"
The size of the import,
.Fa afunc
may
.Em increase
this size if that is desirable, but must never decrease it.
.It Fa size_t
The desired alignment of the imported space.
.It Fa int
the
```

```
.Fa vmflag
argument used for the import.
.El
#endif /* ! codereview */
.It Fa ffunc
A function which is called to return spans to
.Fa source ,
which may be
.Fa source .
Which may be
.Dv NULL
if this arena does not import from another.
This is a
.Vt vmem_free_t
a function taking three parametes and returning void:
.Bl -tag -width Ds
.It Fa "vmem_t"
The arena to which space is being returned.  The
.Fa source
argument to
.Fn vmem_create
or
.Fn vmem_xcreate .
.It Fa "void *"
The span being returned to the source arena.
.It Fa "size_t"
The size of the span being returned to the source arena.
.El
#endif /* ! codereview */
.It Fa source
An arena from which this arena will import,
which may be
An arena from which this arena will import.
Which may be
.Dv NULL
if this arena does not import from another.
.It Fa qcache_max
Each arena offers caching of integer multiples of
.Fa quantum
up to
.Fa qcache_max ,
which may be 0.
.Fa qcache_max .
.It Fa vmflag
A bitmask of flags indicating the characteristics of this arena.
.Bl -tag -width Ds
.It Dv VMC_IDENTIFIER
The arena represents arbitrary integer identifiers, rather than virtual
memory.
.El
.It Fa vmp
A pointer to the vmem arena to be destroyed.
.El
.Sh DESCRIPTION
A
.Em vmem arena
is a section of an arbitrary address space (a range of integer addresses).
This commonly represents virtual memory, but can in fact be an arbitrary set
of integers. The
.Dv VMC_IDENTIFIER
flag set at arena creation time differentiates between these two cases.
.Pp
The
.Fa afunc ,
.Fa ffunc , and
.Fa source
```

```
arguments combine to support a hierarchical structure of arenas, each
importing from a single parent (the
.Fa source ) .
The
.Fn vmem_create
and
.Fn vmem_xcreate
functions differ in that the latter provides an interface for
.Fa afunc
to alter the size of the span imported from
.Fa source .
It is only legal to
.Em increase
this size.
thise size.
.Sh CONTEXT
These functions can be called from user or kernel context.
.Sh RETURN VALUES
Upon successful completion the
.Fn vmem_create
and
.Fn vmem_create and
.Fn vmem_xcreate
functions return a pointer to a vmem arena.  Otherwise,
.Dv NULL
is returned to indicate the arena could not be created.
.Sh SEE ALSO
.Xr vmem 9 ,
.Xr vmem_add 9F ,
.Xr vmem_alloc 9F
```

```
*************************************************************
    1837 Fri Feb  3 18:31:04 2017
new/usr/src/man/man9f/vmem_walk.9f
code review from Josh and Robert
*************************************************************
.\"
.\" This file and its contents are supplied under the terms of the
.\" Common Development and Distribution License ("CDDL"), version 1.0.
.\" You may only use this file in accordance with the terms of version
.\" 1.0 of the CDDL.
.\"
.\" A full copy of the text of the CDDL should have accompanied this
.\" source.  A copy of the CDDL is also available via the Internet at
.\" http://www.illumos.org/license/CDDL.
.\"
.\"
.\" Copyright 2017, Richard Lowe.
.\"
.Dd Jan 18, 2017
.Dt VMEM_WALK 9F
.Os
.Sh NAME
.Nm vmem_walk ,
.Nm vmem_size
.Nd walk a (sub\-)set of the segments in a vmem arena
.Sh SYNOPSIS
.In sys/vmem.h
.Ft void
.Fo vmem_walk
.Fa "vmem_t *vmp"
.Fa "int typemask"
.Fa "void (*func)(void *, void *, size_t)"
.Fa "void *arg"
.Fc
.Ft size_t
.Fo vmem_size
.Fa "vmem_t *vmp"
.Fa "int typemask"
.Fc
.Sh INTERFACE LEVEL
illumos DDI specific
.Sh PARAMETERS
.Bl -tag -width Ds
.It Fa vmp
The vmem arena to walk.
The vmem arena to walk
.It Fa typemask
A bitmask indicating the types of segment to operate on.
A bitmask indicating the types of segment to operate on
.Bl -tag -width Ds
.It Dv VMEM_ALLOC
Allocated segments.
Allocated segments
.It Dv VMEM_FREE
Free segments.
Free segments
.El
.It Fa func
The function to apply to each segment matching
.Fa typemask .
.Fa func
should accept 3 arguments and return void:
should be of 3 values and return void:
.Bl -tag -width Ds
.It Fa "void *arg"
The
```

```
.Fa arg
passed to
.Fn vmem_walk .
.Fn vmem_walk
.It Fa "void *vaddr"
The base address of the segment.
The base address of the segment
.It Fa "size_t size"
The size of the segment.
The size of the segment
.El
.It Fa arg
An arbitrary argument passed to each call to
.Fn func .
.Fn func
.El
.Sh DESCRIPTION
.Fn vmem_walk
walks each segment in the arena
.Fa vmp
and applies
.Fa func
to each which matches
.Fa typemask .
.Pp
.Fn vmem_size
walks each segment in the arena
.Fa vmp
and totals the size of each matching
.Fa typemask .
.Sh CONTEXT
This function may be called from user or kernel context.
.Sh SEE ALSO
.Xr vmem 9 ,
.Xr vmem_alloc 9F ,
.Xr vmem_create 9F
```