Glossary

ABI
Application Binary Interface.

adjmsg
Trim bytes from a message.

allocb
Allocate a message block.

ASSERT
Program verification macro.

badaddr
Check for bus error when reading an address.

bcanput
Test for flow control in a specified priority band.

bcopy
Copy data between address locations in the kernel.

big-endian
The default for a byte order.

biodone
Release buffer after block I/O and wakeup processes.

bioerror
Manipulate error field within a buffer header.

biowait

Suspend processes pending completion of block I/O.

block driver

A device driver, such as for magnetic tape or disk drives, that transfers data in blocks through the buf structure.

bp_mapin

Allocate virtual address space for buffer page list.

bp_mapout

Deallocate virtual address space for buffer page list.

brelse

Return a buffer to the system's free list.

btod

Convert from bytes to disk sectors.

btop

Convert size in bytes to size in pages (rounded down).

bptophys

Get physical address of buffer data.

btopr

Return number of memory pages contained in the specified number of bytes, rounded up.

buf

Block I/O data transfer structure, the basic data structure for block I/O transfers.

bufcall

Call a function when a buffer becomes available.

bus-watching cache

When an IP5, IP7, or IP19 system performs a DMA write into physical memory, the bus-watching cache automatically invalidates the data cache. This hardware function eliminates the need for data cache write back or invalidation in software.

bzero

Clear memory for a given number of bytes.

canput

Test for flow control in a stream.

character device

A device driver, such as a terminal or printer, that transfers data character by character. See also block device.

character driver

A device driver, such as for a terminal or printer, that transfers data characters between the device and the user program. Note that block devices, such as magnetic tape or disk drives, also support character access.

close

Relinquish access to a device. The user process invokes the **close**() system call when it is finished with a device, but the system does not necessarily execute your *drv***close**() entry point for that device.

clrbuf

Erase the contents of a buffer.

cmn_err

Display an error message or panic the system.

copyb

Copy a message block.

copyin

Copy data from user process virtual space to kernel virtual space.

copymsg

Copy a message.

copyout

Copy data from kernel virtual space to user process virtual space.

copyreq

STREAMS transparent **ioctl**() copy request structure – data necessary to process transparent ioctls.

copyresp

STREAMS transparent **ioctl**() copy response – data in response to a prior copy request necessary to continue processing transparent ioctls.

cpsema

Conditionally perform a "P" or wait semaphore operation.

cvsema

Conditionally perform a "V" or wait semaphore operation.

data structure

The memory storage area used to hold data types such as integers, strings, or an array of integers. The data structures associated with drivers are used as buffers for holding data being moved between the user data area and the device.

datab

STREAMS data block structure that describes the data of a STREAMS message.

datamsg

Test whether a message is a data message.

DDI/DKI

Device Driver Interface/Device Kernel Interface.

delay

Delay process execution for a specified number of clock ticks.

devflag

Driver flags – Silicon Graphics only supports flags D_MP, D_WBACK and D_OLD device.

device driver

A software routine that manages a hardware device; it brings the device into and out of service, sets hardware parameters in the device, transmits data from the kernel to the device, receives data from the device and passes data back to the kernel, and handles I/O errors.

Device Driver Interface (DDI)

The set of structures, routines, and optional functions used to implement a device driver.

device types

There are two types of devices available on any UNIX system: software and hardware. A software device is usually a section of memory and is referred to as a pseudo-device. A pseudo-device may provide access to system structures that are unavailable at the user level. For example, a pseudo-device such as a RAM disk could provide fast access to files. Some examples of hardware devices are disk drives, tape drives, printers, scanners, and terminals.

dki_dcache_inval

Invalidate the data cache for a given range of virtual addresses.

dki_dcache_wb

Write back the data cache for a given range of virtual addresses.

dki_dcache_wbinval

Write back and invalidate the data cache for a given range of virtual addresses.

dma_map

Load DMA mapping registers for an imminent transfer.

dma_mapaddr

Return the "bus virtual" address for a given map and address.

dma_mapalloc

Allocate a DMA map. See the dma_map(D3X) man page.

dma_mapfree

Free a DMA map. See the dma_map(D3X) man page.

downstream

The direction of STREAMS messages flowing through a write queue from the user process to the driver.

Driver-Kernel Interface (DKI)

A defined service interface for the entry point routines and utility functions specified for communications between the driver and the kernel. It does not include the driver/hardware or the driver/boot software interface.

drv_getparm

Retrieve kernel state information

drv_hztousec

Convert clock ticks to microseconds.

drv_priv

Determine whether credentials are privileged.

drv_setparm

Set kernel state information.

drv_usectohz

Convert microseconds to clock ticks.

drv_usecwait

Busy-wait for specified interval.

dupb

Duplicate a message block.

dupmsg

Duplicate a message.

edtinit

Initialize a device at boot time.

EISA bus

Enhanced Industry Standard Architecture bus.

EISA Product Identifier (ID)

EISA expansion boards have a four-byte product identifier (z=0 for the system board).

eisa_dma_disable

Disable recognition of hardware requests on a DMA channel.

eisa_dma_enable

Enable recognition of hardware requests on a DMA channel.

eisa_dma_free_buf

Free a previously allocated DMA buffer descriptor.

eisa_dma_free_cb

Free a previously allocated DMA command block.

eisa_dma_get_buf

Allocated DMA buffer descriptor.

eisa_dma_get_cb

Allocated a DMA command block.

eisa_dma_prog

Program a DMA operation for a subsequent software request.

eisa_dma_stop

Stop software-initiated DMA operation on a channel and release it.

eisa_dma_swstart

Initiate a DMA operation via software request.

enableok

Allow a queue to be serviced.

Enhanced Industry Standard Architecture

The EISA bus specification.

errnos

Error numbers.

esballoc

Allocate a message block using an externally supplied buffer.

esbbcall

Call a function when an externally supplied buffer can be allocated.

etoimajor

Convert external to internal major device number.

flushband

Flush messages in a specified priority band.

flushbus

Make sure contents of the write buffer are flushed to the system bus.

flushq

Flush messages on a queue.

freeb

Free a message block.

freemsg

Free a message.

freerbuf

Free a raw buffer header.

freesema

Free the resources associated with a semaphore.

free_rtn
STREAMS driver's message free routine structure.

fubyte
Fetch (read) a byte from user space.

fuword
Fetch (read) a word from user space.

geteblk
Get an empty buffer.

getemajor
Get external major device number.

geteminor
Get external minor device number.

geterror
Retrieve error number from a buffer header.

getmajor
Get internal major device number.

getminor
Get internal minor device number.

getq
Get the next message from a queue.

getrbuf
Get a raw buffer header.

GIO bus
Graphics I/O bus used on Indigo, Indigo^2, and Indy workstations.

halt
Shut down the driver when the system shuts down.

I/O operations

Services that provide access to shared input/output devices and to the global data structures that describe their status. I/O operations open and close files and devices, read data from and write data to devices, set the state of devices, and read and write system data structures.

info

STREAMS driver and module information.

init

Initialize a device.

initnsema

Allocate a semaphore and initialize it to a given value.

insq

Insert a message into a queue.

inter-process communication

These are system calls that allow a process to send information to another process. There are several ways of sending information to another process: signals, pips, shared memory, message queues, semaphores, or streams and sockets.

interrupt level

A driver interrupt routine that is started when an interrupt is received from a hardware device. The system accesses the interrupt vector table, determines the major number of the device, and passes control to the appropriate interrupt routine.

interrupt priority level

The interrupt priority level at which the device requests that the CPU call an interrupt process. This priority can be overridden in the drivers's interrupt routine for critical sections of code with the spl function.

intr

Process a device interrupt after a transfer terminates (either normally upon completion or abnormally due to some error).

iocblk

STREAMS ioctl structure.

ioctl

Control a character device. Character devices may include a "special function" entry point, *drv***ioct**().

iovec

Data storage structure for I/O using uio.

IRQ

See Interrupt Request Input.

itimeout

Execute a function after a specified length of time.

itoemajor

Convert internal to external major device number.

k0

Virtual address range that is cached but not mapped by translation look-aside buffers. Also *kseg0*.

k1

Virtual address range that is neither cached nor mapped. Also *kseg1*.

k2

Virtual address range that can be both cached and mapped by translation look-aside buffers. Also *kseg2*.

kern_calloc

Allocate storage for objects of a specified size.

kern_free

Free kernel memory space

kern_malloc

Allocate kernel virtual memory.

kmem_alloc

Allocate space from kernel free memory.

kmem_free

Free previously allocated kernel memory.

kmem_zalloc

Allocate and clear space from kernel free memory.

kvtophys

Get physical address of buffer data.

linkb

Concatenate two message blocks.

linkblk

STREAMS multiplexor link structure – data needed by a multiplexing driver to set up or take down a multiplexor link.

LOCK

Acquire a basic lock Silicon Graphics LOCK function returns int instead of pl_t.

LOCK_ALLOC

Allocate and initialize a basic lock. Silicon Graphics doesn't support compilation option _LOCKTEST. splockmeter is provided for debugging purpose by Silicon Graphics.

LOCK_DEALLOC

Deallocate an instance of a basic lock

makedevice

Make device number from major and minor numbers.

map

Support virtual mapping for memory-mapped device.

max

Return the larger of two integers.

messages

STREAMS messages.

min

Return the lesser of two integers.

mmap

Check virtual mapping for memory-mapped device. (Silicon Graphics also supports map and unmap entry routines).

mmapped device driver

Memory-mapped device drivers are those in which the hardware is memory mapped into a user's address space; no interrupt or DMA service routine is available to the user process.

module

A STREAMS module consists of two related queue structures, one for upstream messages and one for downstream messages. One or more modules may be pushed onto a stream between the stream head and the driver, usually to implement and isolate a communication protocol or a line discipline.

module_info

STREAMS driver and module information – identification and limit values used to initialize the module's or driver's queues.

msgb

STREAMS message block structure.

msgdsize

Return number of bytes of data in a message.

ngeteblk

Get an empty buffer of the specified size.

noenable

Prevent a queue from being scheduled.

open

Gain access to a device. The kernel calls drvopen() when the user process issues an **open**() system call.

OTHERQ

Get pointer to queue's partner queue.

pcmsg

Test whether a message is a priority control message.

phalloc

Allocate and initialized a pollhead structure.

phfree

Free a pollhead structure.

physiock

Validate and issue raw I/O request.

PIO

Programmed I/O.

pio_badaddr

Check for bus error when reading an address.

pio_bcopyin

Copy data from VME bus address to kernel's virtual space.

pio_bcopyout

Copy data from kernel's virtual space to VME bus address.

pio_mapaddr

Used with FIXED maps to generate a kernel pointer to VME bus space.

pio_mapalloc

Allocate a PIO map.

pio_mapfree

Free up a previously allocated PIO map.

pio_wbadaddr

Check for bus error when writing to an address.

poll

Poll entry point for a non-stream character driver. Silicon Graphics currently does not support **POLLRDNORM**, **POLLWRNORM**, **POLLRDBAND**, and **POLLWRBAND**. A character device driver may include a *drv***poll**() entry point so that users can use **select**(2) or **poll**(2) to poll the file descriptors opened on such devices.

pollwakeup

Inform polling processes that an event has occurred.

prefix

Driver prefix. Throughout this manual, the prefix *drv* preceding a function, routine, or entry point represents the name of the device driver you are writing.

primatives

C operations from which more complex operations can be constructed.

print

Display a driver message on the system console.

process control

These are system calls that allow a process to control its own execution. A process can allocate memory, lock itself in memory, set its scheduling priorities, wait for events, execute a new program, or create a new process.

proc_ref

Obtain a reference to a process for signaling.

proc_signal
Send a signal to a process.

proc_unref
Release a reference to a process.

psema
Perform a "P" or wait semaphore operation.

pseudo-device
A section of memory that emulates the functionality of a hardware device in software. Pseudo-devices may provide access to system structures that are unavailable at the user level. For example, a pseudo-device such as a RAM disk could provide fast access to files.

ptob
Convert size in pages to size in bytes.

put
Receive messages from the preceding queue.

putbq
Place a message at the head of a queue.

putctl
Send a control message with a one-byte parameter to a queue.

putctl1
Send a control message with a one-byte parameter to a queue.

putnext
Send a message to the next queue.

putq
Put a message on a queue.

qenable
Schedule a queue's service routine to be run.

qinit

STREAMS queue initialization structure – pointers to processing procedures and default values for a **queue**().

qreply

Send a message in the opposite direction in a stream.

qsize

Find the number of messages on a queue.

queue

STREAMS queue structure – pointers to processing procedures, the next queue in the stream, flow control parameters, and messages.

RD

Get a pointer to the read queue.

read

Read data from a device. The kernel executes the drvread() or drvwrite() entry points whenever a user process calls the **read**() system call.

rmalloc

Allocate space from a private space management map.

rmallocmap

Allocate and initialize a private space management map.

rmalloc_wait

Allocate space from a private space management map.

rmfree

Free space into a private space management map.

rmfreemap

Free private space management map.

rmvb

Remove a message block from a message.

rmvq

Remove a message from a queue.

SAMESTR

Test whether the next queue is of the same type.

SCSI

Small Computer System Interface.

SCSI bus

See Small Computer System Interface.

SCSI driver interface

A collection of machine-independent input/output controls, functions, and data structures, that provide a standard interface for writing a SCSI driver.

scsi_alloc

Allocate communication channel between host adapter driver and a kernel level SCSI device driver

scsi_command

Issue a command to a SCSI device

scsi_free

Free communication channel between host adapter driver and a kernel level SCSI device driver

scsi_info

Get information about a SCSI device

sgset

Assign physical addresses to a vector of software scatter/gather registers.

signals

Signal numbers.

size

Return size of logical block device.

sleep

Suspend process execution pending occurrence of an event.

SLEEP_ALLOC

Allocate and initialize a sleep lock Silicon Graphics doesn't support compilation option _MPSTATS.

SLEEP_DEALLOC

Deallocate an instance of a sleep lock.

SLEEP_LOCK

Acquire a sleep lock. Always pass -1 as priority.

```
void SLEEP_LOCK(sleep_t *lockp, -1)
```

SLEEP_LOCKAVAIL

Query whether a sleep lock is available.

SLEEP_LOCK_SIG

Acquire a sleep lock The valid values for priority are as follows: PUSER, PCATCH, PSLEP, PPIPE, PVFS, and PWAIT SLEEP_TRYLOCK Try to acquire a sleep lock.

SLEEP_TRYLOCK

Try to acquire a sleep lock.

SLEEP_UNLOCK

Release a sleep lock.

socket

A software structure that represents one endpoint in a two-way communications link. Created by **socket**(2).

spl

Block/allow interrupts on a processor.

srv

Service queued messages.

start

Start initialize a device at system start-up.

strategy

Perform block I/O strategy.

strcat

Concatenate strings.

strcpy

Copy a string.

Stream

A linked list of kernel data structures that provide a full-duplex data path between a user process and a device. Streams are supported by the STREAMS facilities in UNIX System V Release 3 and later.

stream head

The stream head, which is inserted by the STREAMS subsystem, processes STREAMS-related system calls and performs data transfers between user space and kernel space. It is the component of a stream closet to the user process. Every stream has a stream head.

STREAMS

A kernel subsystem used to build a stream, which is a modular, full-duplex data path between a device and a user process. In IRIX 5.x and later, the TCP/IP stack sits on top of the STREAMS stack. The Transport Layer Interface (TLI) is fully supported.

streamstab

STREAMS driver and module declaration structure.

streams_interrupt

Synchronize interrupt-level function with STREAMS mechanism.

STREAMS_TIMEOUT

Synchronize timeout with STREAMS mechanism.

strlog

Submit messages to the log driver.

stroptions

STREAMS head option structure.

strqget

Get information about a queue or band of the queue.

strqset

Change information about a queue or band of the queue.

subyte

Set (write) a byte to user space.

suword

Set (write) a word to user space.

TCP/IP

Transmission Control Protocol/Internet Protocol.

TFP

SGI's pre-release, internal code name for the MIPS R8000 processor.

TLI

Transport Interface Layer.

TRYLOCK

Try to acquire a basic lock.

uio

scatter/gather I/O request – describes an I/O request that can be broken into different data storage areas.

uiomove

Copy data using uio structure.

uiophysio
Set up user data space for I/O.

unbufcall
Cancel a pending bufcall request.

undma
Unlock physical memory in user space.

unlinkb
Remove a message block from the head of a message.

unload
Clean up a loadable kernel module.

UNLOCK
Release a basic lock

unmap
Support virtual unmapping for memory-mapped device

untimeout
Cancel previous timeout request.

untimeout_func
Cancel a previous invocation of timeout by function.

ureadc
Copy a character to space described by uio structure.

userdma
Lock, unlock physical memory in user space

uwritec
Return a character from space described by uio structure.

valusema
Return the value associated with a semaphore.

VME bus

VERSA Module Eurocard bus.

VME-bus adapter

A hardware conduit that translates host CPU operations to VME-bus operations and decodes some VME-bus operations to translate them to the host side.

vme_adapter

Determine VME adapter.

vme_ivec_alloc

Allocate a VME bus interrupt VECTOR.

vme_ivec_free

Free up a VME bus interrupt VECTOR.

vme_ivec_set

Register a VME bus interrupt handler.

volatile

Inform the compiler of volatile variables.

vpsema

Perform an atomic "V" and "P" semaphore operation on two semaphores.

vsema

Perform a "V" or signal semaphore operation.

v_getaddr

Get the user address associated with virtual handle.

v_gethandle

Get unique identifier associated with virtual handle.

v_getlen

Get length of user address space associated with virtual handle.

v_mapphys

Map physical addresses into user address space.

wakeup

Resume suspended process execution.

wbadaddr

Check for bus error when writing to an address.

WR

Get a pointer to the write queue.

write

Write data to a device. The kernel executes the drvread() or drvwrite() entry points whenever a user process calls the **read**() or **write**() system calls.