

The Linux Kernel API

The Linux Kernel API

This documentation is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

For more details see the file COPYING in the source distribution of Linux.

Table of Contents

1. Driver Basics.....	1
1.1. Driver Entry and Exit points	1
module_init	1
module_exit.....	2
1.2. Atomic and pointer manipulation	2
atomic_read.....	3
atomic_set	3
atomic_add.....	4
atomic_sub	5

Chapter 1. Driver Basics

1.1. Driver Entry and Exit points

module_init

Name

`module_init` — driver initialization entry point

Synopsis

```
module_init ( x );
```

Arguments

`x`

function to be run at kernel boot time or module insertion

Description

`module_init` will add the driver initialization routine in the “`__initcall.int`” code segment if the driver is checked as “y” or static, or else it will wrap the driver initialization routine with `init_module` which is used by `insmod` and `modprobe` when the driver is used as a module.

module_exit

Name

`module_exit` — driver exit entry point

Synopsis

```
module_exit ( x );
```

Arguments

x

function to be run when driver is removed

Description

`module_exit` will wrap the driver clean-up code with `cleanup_module` when used with `rmmod` when the driver is a module. If the driver is statically compiled into the kernel, `module_exit` has no effect.

1.2. Atomic and pointer manipulation

atomic_read

Name

`atomic_read` — read atomic variable

Synopsis

```
atomic_read ( v );
```

Arguments

`v`

pointer of type `atomic_t`

Description

Atomically reads the value of `v`. Note that the guaranteed useful range of an `atomic_t` is only 24 bits.

atomic_set

Name

`atomic_set` — set atomic variable

Synopsis

```
atomic_set ( v ) i i ;
```

Arguments

v

pointer of type `atomic_t`

i

required value

Description

Atomically sets the value of *v* to *i*. Note that the guaranteed useful range of an `atomic_t` is only 24 bits.

atomic_add

Name

`atomic_add` — add integer to atomic variable

Synopsis

```
void atomic_add (int i); atomic_t * v);
```

Arguments

i

integer value to add

v

pointer of type `atomic_t`

Description

Atomically adds *i* to *v*. Note that the guaranteed useful range of an `atomic_t` is only 24 bits.

atomic_sub

Name

`atomic_sub` — subtract the atomic variable

Synopsis

```
void atomic_sub (int i); atomic_t * v);
```

Arguments

i