The /proc/sequence module source
[Posted February 10, 2003 by corbet]

```
/*
 * Simple demonstration of the seq_file interface.
 *
 * $Id: seq.c,v 1.1 2003/02/10 21:02:02 corbet Exp $
 */

#include <linux/init.h>
#include <linux/module.h>
#include <linux/proc_fs.h>
#include <linux/fs.h>
#include <linux/seq_file.h>
#include <linux/slab.h>

MODULE_AUTHOR("Jonathan Corbet");
MODULE_LICENSE("Dual BSD/GPL");


/*
 * The sequence iterator functions.  We simply use the count of the
 * next line as our internal position.
 */
static void *ct_seq_start(struct seq_file *s, loff_t *pos)
{
        loff_t *spos = kmalloc(sizeof(loff_t), GFP_KERNEL);
        if (! spos)
                return NULL;
        *spos = *pos;
        return spos;
}

static void *ct_seq_next(struct seq_file *s, void *v, loff_t *pos)
{
        loff_t *spos = (loff_t *) v;
        *pos = ++(*spos);
        return spos;
}

static void ct_seq_stop(struct seq_file *s, void *v)
{
        kfree (v);
}

/*
 * The show function.
 */
static int ct_seq_show(struct seq_file *s, void *v)
{
        loff_t *spos = (loff_t *) v;
        seq_printf(s, "%Ld\n", *spos);
        return 0;
```

ользование файлов-

```
}

/*
 * Tie them all together into a set of seq_operations.
 */
static struct seq_operations ct_seq_ops = {
        .start = ct_seq_start,
        .next  = ct_seq_next,
        .stop  = ct_seq_stop,
        .show  = ct_seq_show
};

/*
 * Time to set up the file operations for our /proc file.  In this case,
 * all we need is an open function which sets up the sequence ops.
 */

static int ct_open(struct inode *inode, struct file *file)
{
        return seq_open(file, &ct_seq_ops);
};

/*
 * The file operations structure contains our open function along with
 * set of the canned seq_ ops.
 */
static struct file_operations ct_file_ops = {
        .owner   = THIS_MODULE,
        .open    = ct_open,
        .read    = seq_read,
        .llseek  = seq_lseek,
        .release = seq_release
};

/*
 * Module setup and teardown.
 */

static int ct_init(void)
{
        struct proc_dir_entry *entry;

        entry = create_proc_entry("sequence", 0, NULL);
        if (entry)
                entry->proc_fops = &ct_file_ops;
        return 0;
}

static void ct_exit(void)
{
```

```
        remove_proc_entry("sequence", NULL);
}

module_init(ct_init);
module_exit(ct_exit);
```