Consider the following loadable kernel module (named procfs2.c) using the /proc interface. Assume the name of the driver is procbuf2.

/\*

\* procfs2.c - create a "file" in /proc

\*/

#include <linux/kernel.h> /\* We're doing kernel work \*/

#include <linux/module.h> /\* Specifically, a module \*/

#include <linux/proc\_fs.h> /\* Necessary because we use the proc fs \*/

#include <linux/uaccess.h> /\* For copy\_from\_user \*/

#include <linux/version.h>

#include <linux/string.h> /\* For memset \*/

#include <linux/slab.h> /\* for kmalloc \*/

#define PROCFS\_MAX\_SIZE 1024

#define PROCFS\_NAME "procbuf2"

/\* This structure hold information about the /proc file \*/

static struct proc\_dir\_entry \*our\_proc\_file;

/\* The buffer used to store character for this module \*/

static char procfs\_buffer[PROCFS\_MAX\_SIZE] = "HelloWorld\n";

/\* The size of the buffer \*/

static unsigned long procfs\_buffer\_size = 0;

/\* This function is called then the /proc file is read \*/

static ssize\_t procfile\_read(struct file \*file\_pointer, char \_\_user \*buffer,

size\_t buffer\_length, loff\_t \*offset)

{

int len = sizeof(procfs\_buffer);

ssize\_t ret = len;

if (\*offset >= len) {

return 0;

}

if (copy\_to\_user(buffer, procfs\_buffer, len)) {

ret = 0;

} else {

\*offset += len;

}

return ret;

}

/\* This function is called with the /proc file is written. \*/

static ssize\_t procfile\_write(struct file \*file, const char \_\_user \*buff,

size\_t len, loff\_t \*off)

{

procfs\_buffer\_size = len;

if (procfs\_buffer\_size > PROCFS\_MAX\_SIZE)

procfs\_buffer\_size = PROCFS\_MAX\_SIZE;

\*off += procfs\_buffer\_size;

pr\_info("procfile write %s\n", procfs\_buffer);

return procfs\_buffer\_size;

}

static const struct proc\_ops proc\_file\_fops = {

.proc\_read = procfile\_read,

.proc\_write = procfile\_write,

};

static int \_\_init procfs2\_init(void)

{

our\_proc\_file = proc\_create(PROCFS\_NAME, 0644, NULL, &proc\_file\_fops);

if (NULL == our\_proc\_file) {

pr\_info("Error:Could not initialize /proc/%s\n", PROCFS\_NAME);

return -ENOMEM;

}

pr\_info("/proc/%s created\n", PROCFS\_NAME);

return 0;

}

static void \_\_exit procfs2\_exit(void)

{

proc\_remove(our\_proc\_file);

pr\_info("/proc/%s removed\n", PROCFS\_NAME);

}

module\_init(procfs2\_init);

module\_exit(procfs2\_exit);

MODULE\_LICENSE("GPL");

What is the terminal output (not dmesg) of the following commands:

1. cat /proc/procbuf2
2. echo "Cougs" > /proc/procbuf2 && cat /proc/procbuf2

A computer screen shot of a problem solution

Description automatically generated