CS333 - Project 1

Group information

Nguyen Ho Huu Nghia 1751013 - **17APCS2**

Dao Hieu 1751005 - **17APCS1**

Nguyen Truong Vinh Thuyen 1751042 - **17APCS1**

How to use

Compile and load the module to the kernel

```
cd ./random-number-char-dev-driver
make all
```

```
sudo rmmod random_number_driver
```

```
sudo insmod random_number_driver.ko
```

```
dmesg # see if the module is loaded
```

```
sudo chmod 666 /dev/random_number_char_dev
```

Everytime, you read the file, a new random number is returned

The string returned has 11 characters, if the random number does not take up 11 characters, space is padded at the front

- An example way to run is through `cat`

```
sudo cat /dev/random_number_char_dev
```

Modules

Random function

Random function random a signed integer and convert it to a char array and return it.

```
char *random(void)
{
    const int MAX_LENGTH = 11;
```

```
    int i;
    // generate random number as integer
    get_random_bytes(&i, sizeof(i));
    printk("%d", i);

    // count the number of digits (including '-')
    int t = i, c = 0;
    if (t < 0) {
        t = -t;
        c = 1;
    }
    while (t > 0) {
        t = t / 10;
        ++c;
    }

    // allocate string
    char *str = kzalloc(MAX_LENGTH, GFP_KERNEL);
    int index = 0;

    // pad space at the start
    int remaining = MAX_LENGTH - c, j = 0;
    while (j++ < remaining) {
        str[index++] = ' ';
    }

    // add '-' (if applicable)
    if (i < 0) {
        str[index] = '-';
        i = i * -1;
        ++index;
    }

    // extract the rest of the digits (these digits are in the wrong direction)
    char *temp = kzalloc(c + 1, GFP_KERNEL);
    int temp_index = 0;
    while (i > 0) {
        temp[temp_index] = (char)(i % 10 + '0');
        i = i / 10;
        ++temp_index;
    }

    // revert the digits to the right direction
    int k = temp_index - 1;
    while(k >= 0) {
        str[index++] = temp[k--];
    }

    printk("Final random number: %s", str);
    return str;
}
```

Random number generator character device driver

The source code is in ./random-number-char-dev-driver - `Kbuild` + `Makefile` : keep the instructions to compile `random_number_driver.c` and `random_number_driver.h` into loadable object files - `random_number_driver.h` : contains configuration for the character device. *More details can be found in the comments in the source code*. - `random_number_driver.c` : the main code for the driver. The overall structure of the file is going to be presented below. However, for more detailed view of the operations inside each funciton, please refer to the comments in the source code. - Structs: - vchar_dev: containing the registers as char arrays for the character device. - vchar_driver: containing the data structures used for the driver - Device specific block: codes instructing the CPU to interact with the character device - OS specific block: codes interacting with the OS to allocate resources and register the driver's entry point to the OS.