ECE 4730: Embedded Systems II

Project 6: Character Device Driver Development

Objectives

* Utilize virtual memory mapping
* Create a Device Driver
* Instantiate the Device Driver
* Interface with the Device Driver as a file

Deliverables

*	Chardev.ko	5
*	Multiplier.ko	10
*	devtest	5
*	Answers to the questions	5
Total:		25

Tips

Save a copy of all files on the SD card in an easily-accessible place on your computer. If there are mounting problems, format it and replace the files.

Procedure

Part 1- Char Device Driver (my_chardev_mem.ko)

1. Make a directory for lab 6. Place the my_chardev_mem.c and my_chardev_mem.h in it.
2. Add <linux/slab.h> to my_chardev_mem.h (it is required for kmalloc and kfree).
3. Copy the Makefile from lab5 and change it to create my_chardev_mem.ko.
4. Run the command 'Make ARCH=arm CROSS_COMPILE=arm-none-eabi-' to compile the character device driver.

Part 2- Testing the Device from User Space (chartest)

1. Copy the file chartest.c from the lab 6 folder and place it in the same folder as my_chardev_mem.h.
2. Run the following command from the same folder:
   arm-linux-gnueabi-gcc -o chartest chartest.c
3. Copy my_chardev_mem.ko and chartest to the SD card. Then transfer the SD card and mount it.
4. Insert the new module (my_chardev_mem.ko), and then read the output instructions. The filename may be changed to whatever you want, but in order for chartest to work, it needs to match the file name given in the c code.
5. Run the modified instruction, then run ./chartest. What happens? What does chartest do?

Part 3- Your Turn (multiplier.ko, memory mapping, device drivers)

1. Copy my_chardev_mem.c and .h to multiplier.c and .h and modify it to write to the multiply IP like it's a file (use memory mapping from multiply.c).
2. You must initialize in the foll0wing order:
   a. Virtual memory mapping
   b. Character device driver
   c. Major number: assigned by Linux
   d. Minor number: assigned 0
3. Handle device registration errors appropriately. You will be graded on this part.
4. Removing must be done inverse of initialization.
5. For the open and close functions, tell the user when device is opened and closed.
6. Write and read:
   a. Read up on 'put_user' and 'get_user'. Consult Linux Device Drivers, 3rd Edition, chapter 3 for tips.
   b. Write will only need to write to addresses 0-7 (reg0,1).

    c. Read will need to read addresses 0-11 (reg0-2) in order to see both input and output.
    d. Both should return number of bytes transferred between user and kernel space.
7. Once compiled, finish the skeleton code in devtest.c and compile it as done for chartest.c
8. Run them and add screenshots.