**Department of Computer Engineering**

**BLG 413E  
System Programming**

**Project 1 Report**

**Muhammed Raşit EROL 150150023**

**Samet Pilav 150160806**

**Joshgun Rzabayli 150160901**

**Spring 2019**

**REPORT**

**Introduction:**

In this project, we are required to implement a character device driver that will play the board game “Master Mind”. In order to implement this project, we modified scull device to achieve goals these are stated in project file. We changed mainly device layout, write operation, read operation and ioctl function.

**Implementation:**

Firstly, we changed the “ioctl” file in order to define our commands as below (see Figure 1). In here, in order to start new game it is needed to pass argument to the function. Hence, we use write operation on this ioctl function. Others perform its duty and return -1, if function fails.

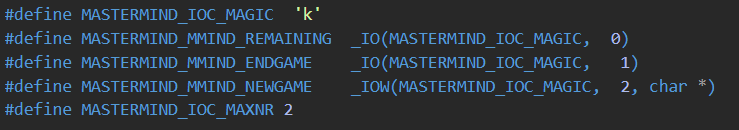


Figure 1 : mastermind\_ioctl.h

After defining commands, we implemented these commands in main file as follow (see Figure 2). In this function remaining part return remaining guess number using our device data structure. End game function calls trim function and it deletes all values in the device. The new game function clears line number for new game and checks given parameter in order get proper argument to the game. For example, 9999 and 999a are not proper parameters, so the function returns -1.

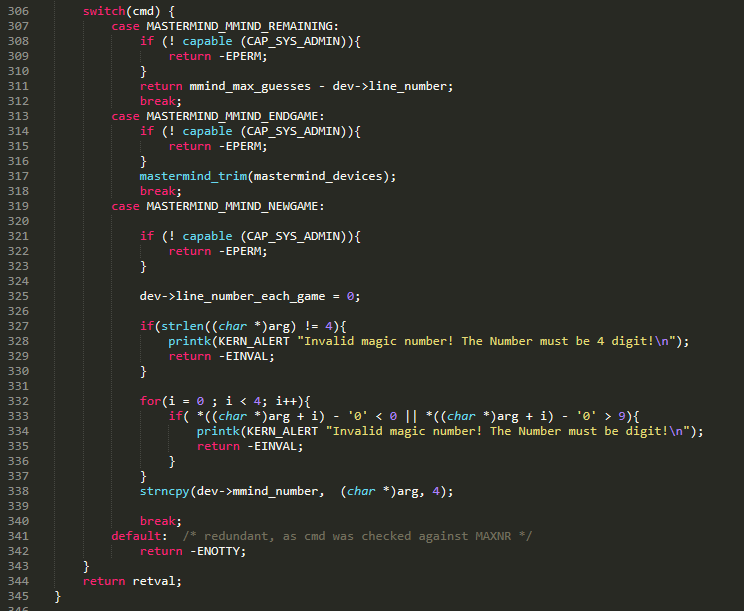


Figure 2 : mastermind\_ioctl function

The Player A has to start the game by defining secret number as in format: “mmind\_number=“4283”” module parameter. The Player B can guess the number as echo “1234” format. The report will be stored in an internal buffer in the form “xxxx m+ n- abcd\n”. For this purpose, we implement structure to hold guess, number of in place digits, number of out of place digits and number of guesses. The line structure as below (see Figure 3):

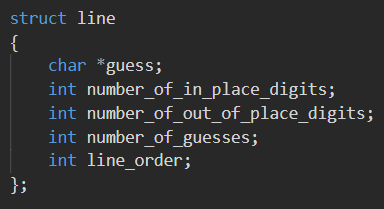


Figure 3 : line struct

The device structure is below (see Figure 4). In here, mmind\_number represent magic number. line\_number variable is used for total number of guesses in order to check game is finished when 256 line is reached. line\_number\_each\_game is used for number of guesses for each game. If new game is started, the variable is assigned to zero as in the ioctl new game function.

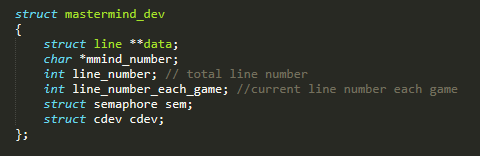


Figure 4: mastermind\_dev struct

The write and read functions are implemented in order perform game requirements. In then write function, mmind\_max\_guesses is used for check maximum number of guesses for each game and also restriction of 256 line is checked with related conditions. Also, input validation is performed in the write function. For each write operation data structure allocate new memory location and used it in order store given parameter during the game flow. The m and n variables described in the project pdf are calculated in the write function.

In the read function, proper conditions are created in order to perform correct reading. The data is read from data structure written in the write function into local buffer. After that, data is sent to read function buffer from local buffer at the end of the function.

In the cleanup function all reserved memory locations is returned back to operating system. Also, in the init module function, major number is taken from system and device is created with default variables.

The test programs written for ioctl functions are also attached the zip file. Some of the program screenshots are added the following part.

**Program Flow:**

In order to active device, it is needed to become root using sudo su command. After that insmod command device is created. In here it is possible to give two parameters to the program. One is mmind\_number which is used for store magic number and its default value is 4283 and mmind\_max\_guesses which is used for maximum number of guesses each game and its default value is 10. The major number which is given us from system can be checked using grep mastermind /proc/devices command. After that using mknod node is created (see Figure 5).

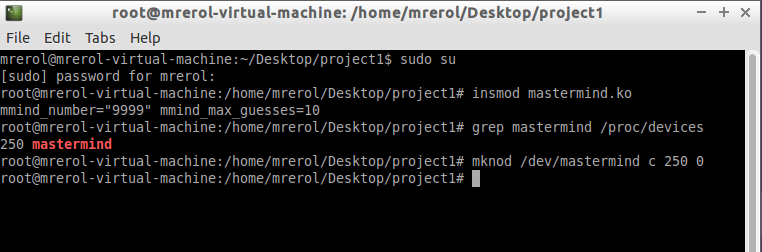


Figure 5 : installation of device

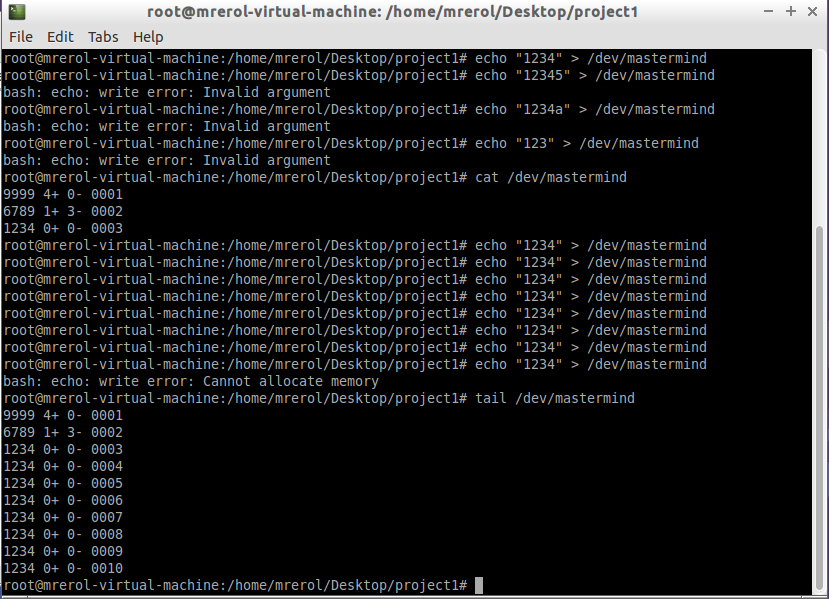


Figure 6: example of program work with writing and reading

In the Figure 6, some of the read and write operations can be seen. Also, some input validation error messages can be seen in there. In the following figures test functions of ioct can be seen (see Figure 7-9).

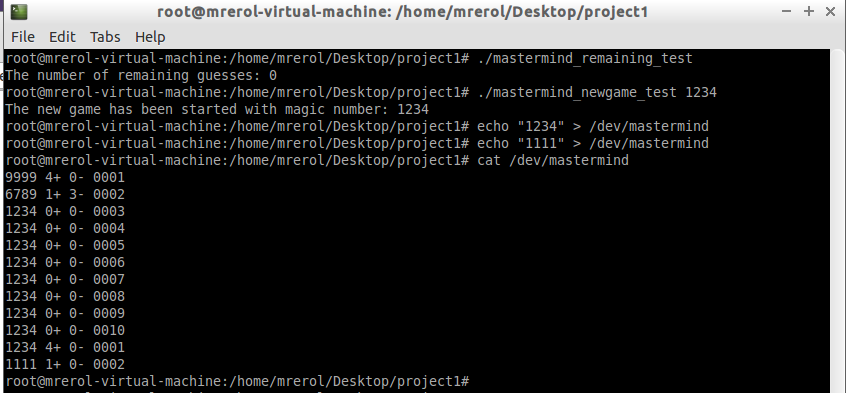


Figure 7: new game and remaining ioctl functions example

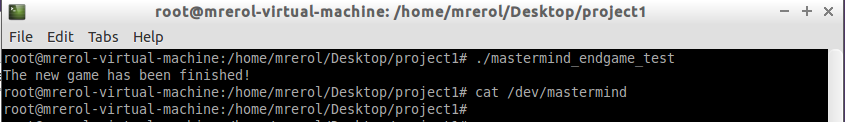


Figure 8: end game test example

In the end of the program, using rmmod and rm -r commands device can be deleted from system (see Figure 9).

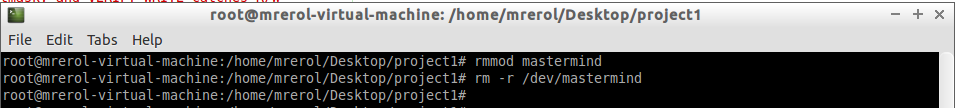


Figure 9: cleaning up the device

In the zip file all source codes and object files of test programs are added. Also, using makefile is attached to the zip file. The program is tested in the Lubuntu 32-bit 14.04 version required in the software\_installation.pdf file.