Wissam Razouki

CSE 4300

Programming Assignment 3

**Parts completed**

* **Part A**
* **Part B**
* **Part C**
* **Part D**
* **Part E**
* **Part F**

**Modifications and Additions**

**Part A:**

* A “kprintf()” statement was modified in “main.c” to print my name whenever I started the kernel.

**Parts B, C, D:**

The modifications/additions made for these parts are very similar to each other.

* A new C source file “simple\_syscalls.c” was created under the “userprog” directory to define the new system calls for parts B, C, D (“\_exit()”, “printint()”, and “reversestring()”, respectively) under the kernel level. Thus, “sys\_” was added before each function.
* The line “file userprog/simple\_syscalls.c” was added to “conf.kern” in order for the kernel to recognize the new system calls when it was reconfigured.
* “sys\_” function prototypes were added to the “syscall.h” header file in order for “syscall.c” to call the new system calls.
* “SYS\_” case statements were added to “syscall.c” to handle the new system calls. The “exit()” function takes “a0” as input, and gives no output. The “printint()” function takes “a0” as input, and gives output to “retval”. The “reversestring()” function takes “a0” and “a1” as inputs, and gives output to “retval”.

In addition, the “err” variable was modified to initialize to 0 in order to avoid error values from constantly appearing in the return variable “v0”.

* New “SYS\_” definitions were added to “callno.h” to support the case statements in “syscall.c”.
* The user-level function prototypes for these system calls were added to “unistd.h” to enable calling the functions from the user-level interface.

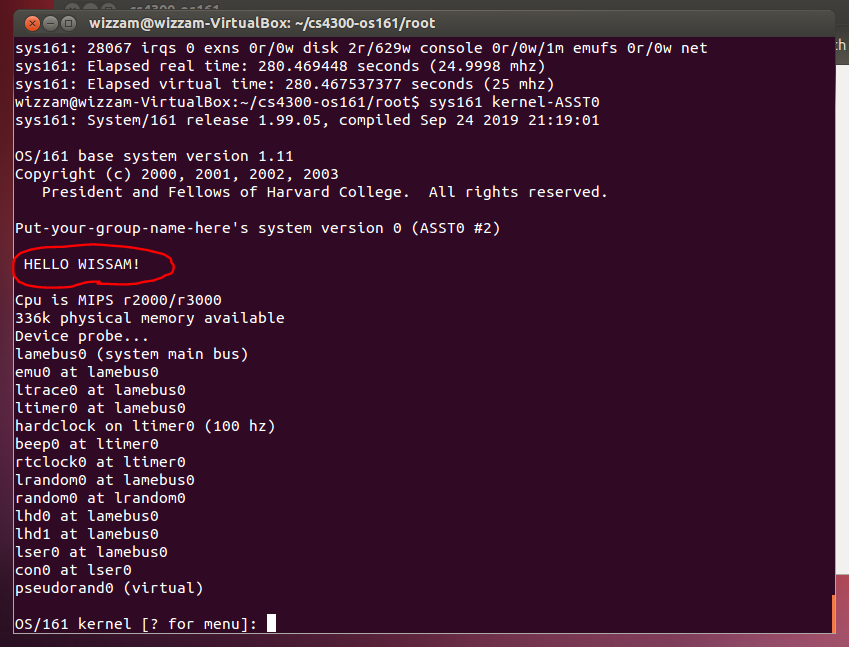
**Parts E, F:**

The modifications/additions made for these parts are very similar to each other.

* “testprint” and “testreverse” directories were created under “testbin” to contain new files for testing the “printint()” and “reversestring()” system calls in the user-level interface.
* For each one, a C source file (“testprint.c” and “testreverse.c”) was written to call the system calls using sample inputs. The return values were captured and displayed using the “printint()” function.
* A “Makefile” and “depend.mk” was copied into each test directory from the “palin” test directory as a sample. These files were modified to support the respective test program.
* “make” statements were added to the “Makefile” under “testbin” directory for the new tests.

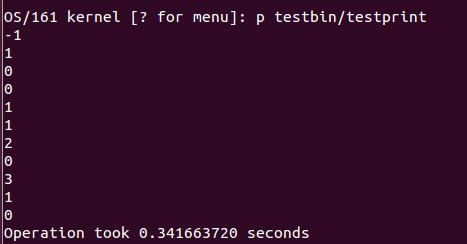
**Sample Outputs**

**Custom greeting:**



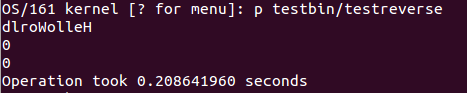
From the image above, we can see that I was able to customize my greeting message to display my name.

**“testprint”:**



This output is for the first test program. The 1st, 3rd, 5th, 7th, and 9th values (-1, 0, 1, 2, 3) are the “kprintf()” outputs from the “printint()” system call, and the 2nd, 4th, 6th, 8th, and 10th values (1, 0, 1, 0, 1) are the return values from the “printint()” function, determined by whether the input integer is a multiple of 2 or not (0 if so, 1 if not). The 11th value (0) is the output given from the “\_exit()” system call with an input of 0.

**“testreverse”:**



This output is for the second test program. The 1st value (“dlroWolleH”) is the “kprintf()” output from the “reversestring()” system call with an input string “HelloWorld” and a string length of 10. The 2nd value (0) is the return value from the “reversestring()” function, determined by whether the input string length is a multiple of 2 or not (0 if so, 1 if not). The 3rd value (0) is the output given from the “\_exit()” system call with an input of 0.