Otto-Friedrich-Universität Bamberg
Fakultät WIAI

Lehrstuhl für Praktische Informatik, insbes.
Systemnahe Programmierung – Prof. Dr. Michael Engel
https://www.uni-bamberg.de/sysnap
michael.engel@uni-bamberg.de

Operating Systems Engineering

Exercises
Summer 2022

Exercise 2

Processes

Discussion on Monday, 13.6.2022

The code skeleton for this exercise can be found at
`https://github.com/michaelengel/OSE2022/tree/main/lecture4/process_swap`
We have a new system call, `exit`, with syscall number 42. This can be used to leave a process in a controlled way. There is also a `Makefile` now that makes building the system easier. You can simply run `make` to build the kernel and two user processes or `make run` to build all code and start qemu. Using `make clean` deletes all intermediary build results (object, ELF and binary files as well as the header files containing the code for the two processes).

2.1 Loadable Processes

Implement the loading of processes.
The memory layout is changed so that the kernel starts at address 0x8000_0000 as before, but there is a new linker script `user.ld` that links the code for user mode processes separately to start at address 0x8010_0000. The `Makefile` compiles the user code separately.
The user process should simply print the character `'a'` and exit using the new system call. After termination of the process, the kernel should reload the process into memory and start it again.
As usual, add code to the sections marked with `TODO`.

You will notice that the system crashes after a number of iterations. Can you figure out why?

2.2 Process switching

Now create **two** user processes, `user1.c` and `user2.c`. Change the `Makefile` and `setup.c` to handle the loading of the two user processes.

The first process should simply print the character `'a'` and exit, the second process should be identical, but print the character `'b'`.

Write a simple **scheduler** inside the exception handling function:
When process 1 terminates, load process 2 into memory and start it.
When process 2 terminates, load process 1 and start it again and so on.

The same problem shows up as with the first task, of course...

Note:
This is one of the first things Linus Torvalds tried on his PC when he started to write what is now known as Linux...