Advanced Linux — Lab 4

Albert Akmukhametov
Email: a.akmukhametov@innopolis.university
Group: BS20-01-CS

Project Structure

```
.
|-- Makefile
|-- src
|   |-- device.c
|   |-- device.h
|   |-- module.c
|   |-- stack.c
|   `-- stack.h
`-- tests
    |-- device.py
    |-- pyproject.toml
    `-- test.py
```

- `Makefile`, obviously, is entry point for everything. It contains the following targets:
  - `all` — building the module
  - `clean` — removing all temporary files generated while building
  - `install` — executes `insmod` on freshly-compiled module
  - `uninstall` — on the other side, executed `rmmod` on the module
  - `test` — running integrating tests
- `src` — source codes of Kernel Module
  - `module.c` — entry point for Kernel Module
  - `device.{c,h}` — the logic responsible for operating as character device
  - `stack.{c,h}` — implementation of stack
- `tests` — integrating tests based on `pytest` framework

Test Report

```
make uninstall && make install && make test
```

```
root@kinjalik-remote:/home/kinjalik/adv-linux-lab4# make uninstall && make install && make test
rmmod stack_dev
insmod stack_dev.ko
pytest tests/test.py
============================================================ test session starts ============================================================
platform linux -- Python 3.8.10, pytest-7.2.0, pluggy-0.13.0
rootdir: /home/kinjalik/adv-linux-lab4/tests, configfile: pyproject.toml
plugins: xdist-1.31.0, forked-1.1.3
collected 8 items

tests/test.py::test_put_delete_by_singles PASSED
tests/test.py::test_put_delete_by_multiples PASSED
tests/test.py::test_partial_write_read PASSED
tests/test.py::test_overflow PASSED
tests/test.py::test_underflow PASSED
tests/test.py::test_basic_resize PASSED
tests/test.py::test_resize_to_smaller_than_actual PASSED
tests/test.py::test_read_after_resize PASSED

============================================================ 8 passed in 0.36s =============================================================
root@kinjalik-remote:/home/kinjalik/adv-linux-lab4#
```

How does it works

Stack

- At initialization we allocate memory for stack of default size (I set it to 5, but in fact it may be any length)
- Any operation (except Init and Uninit) is enclosed between Mutex acquisition and releast to ensure absence of race between some modules
  - **Why not Spin-Locks**: Spinlocks usually used for very-very short waitings (i.e. without I/O operations, without sleeping, so if we waiting for seomeone who is sleeping now, we will just warm the CPU). In my implementation of stack there are logs and memory allocation operations (during controlling the stack size). We can cancel logs, but unfortunately can not cancel memory operations, so it’s better to use usual Mutex
- When size changing operation called, I’ve checked whether operation is suitable (i.e. new buffer can’t store data already lying on stack). If everything is fine, copy from old buffer to new and release old

Autotest

I have wrote some wrappers over read/write/ioctl corresponding to functions of device in order to simplify integrating test.

Here is an example of such wrapper

```
16    def read_nums(dev, cnt: int) -> List[int]:
17        res = os.read(dev, cnt * 4)
18        ret = []
19        for i in range(0, int(len(res) / 4)):
20            print(res[i*4: (i+1) * 4])
21            ret.append(int.from_bytes(res[i * 4: (i+1) * 4], byteorder="little"))
22
23        return ret
24
```

Here is an example of test. Pretty simple :)

```
def test_read_after_resize(open_device):
    dev = open_device
    shorten_size = 3

    assert write_nums(dev, [i for i in range(shorten_size)]) == shorten_size
    assert change_size(dev, shorten_size) == 0

    assert read_nums(dev, shorten_size) == list(reversed([i for i in range(shorten_size)]))

    assert change_size(dev, INITIAL_STACK_SIZE) == 0
```

In fact, also I need at least component tests for Stack (using GTest for example) to ensure it works (also such tests are much cheaper because doesn’t involve kernel mechanisms). But for Assignment purposes I guess this should be enough