Name: Harshit Bansal
Roll No.: 200428

**CS614 Linux Kernel Programming**
**Assignment 3**

## Design Details

- User Program uses the functionality of the device via intermediate library functions.
- The library function relies on two interfaces, namely sysfs and chardev, to communicate with the kernel device driver module. In order to perform certain actions such as creating a handle, closing a handle, setting a key, or configuring the system, the library function makes use of the sysfs file to read or write the required information. On the other hand, when performing encryption or decryption, the library function utilizes the chardev interface to exchange information with the device driver.
- Device Driver Module communicates with the device via MMIO mapping. It reads and writes data to perform encryption and decryption as specified in the device specifications in the Assignment Problem.
- If user passes data of size larger than 16 KB, then it is broken down into 16 KB chunks and driver is called for each chunk independently. This give every process chance to use driver even if a process is encrypting/decrypting large data.

## Implementation Details

- A linked list of all the active handle is kept. The entry of the list consist of fields like the handle_no, keys, configs etc.
- **create_handle** → This function returns handle to the user. This handle is further used by the device as well as the user to set configs and keys and also to do encryption/decryption.
- **close_handle** → This removes the corrsponding handle entry from the list.
- **set_keys, set_configs** → These functions find their entry in the list. If found they set its corresponding fields with the values.
- **encrypt, decrypt** → Since the device handles only one request at a time, inorder to serialize requests from many processes, a mutex needs to be held by the process when doing encryption/decryption on the device. When the encryption/decryption is completed and the message has been read into user buffer the lock is released and the chance is given to other processes waiting for the device to get free.
- **map_card** *to* This function maps the user mmio address space to user address space using the *io_remap_pfn_range*() call. The library mmaps a region and the sends the address to the driver module. The driver module then maps the vm_area to the device mmio address space.

## Test Strategies

- To check for deadlocks, multiple threads were spawned and was made sure that the encryption/decryption is correct.
- To check the case of larger buffer, different file sizes were used for encryption/decryption and was made sure that the encryption/decryption is correct.
- Different configs were also tested.
  - MMIO with Interrupt
  - MMIO with no-Interrupt
  - DMA with Interrupt
  - DMA with no-Interrupt

## Benchmark Results

**MMIO:**

| 10:04:29 PM IST | CPU | %user | %nice | %system | %iowait | %steal | %idle |
|---|---|---|---|---|---|---|---|
| 10:04:31 PM IST | all | 3.73 | 0.00 | 50.75 | 0.00 | 0.00 | 45.52 |
| 10:04:33 PM IST | all | 0.50 | 0.00 | 50.75 | 0.00 | 0.00 | 48.74 |
| 10:04:35 PM IST | all | 0.50 | 0.00 | 51.13 | 0.00 | 0.00 | 48.37 |
| 10:04:37 PM IST | all | 0.50 | 0.00 | 51.00 | 0.00 | 0.00 | 48.50 |
| 10:04:39 PM IST | all | 2.77 | 0.00 | 50.63 | 0.00 | 0.00 | 46.60 |
| 10:04:41 PM IST | all | 1.00 | 0.00 | 51.00 | 0.25 | 0.00 | 47.75 |
| 10:04:43 PM IST | all | 0.25 | 0.00 | 51.25 | 0.00 | 0.00 | 48.50 |

**MMIO Interrupt:**

| 10:11:56 PM IST | CPU | %user | %nice | %system | %iowait | %steal | %idle |
|---|---|---|---|---|---|---|---|
| 10:11:58 PM IST | all | 0.50 | 0.00 | 50.50 | 0.00 | 0.00 | 48.99 |
| 10:12:00 PM IST | all | 0.50 | 0.00 | 50.88 | 0.00 | 0.00 | 48.62 |
| 10:12:02 PM IST | all | 0.25 | 0.00 | 51.25 | 0.25 | 0.00 | 48.25 |
| 10:12:04 PM IST | all | 0.50 | 0.00 | 51.00 | 0.00 | 0.00 | 48.50 |
| 10:12:06 PM IST | all | 2.76 | 0.00 | 51.01 | 0.00 | 0.00 | 46.23 |
| 10:12:08 PM IST | all | 0.50 | 0.00 | 51.01 | 0.00 | 0.00 | 48.49 |
| 10:12:10 PM IST | all | 0.75 | 0.00 | 50.75 | 0.00 | 0.00 | 48.50 |

**DMA:**

| 10:14:10 PM IST | CPU | %user | %nice | %system | %iowait | %steal | %idle |
|---|---|---|---|---|---|---|---|
| 10:14:12 PM IST | all | 0.50 | 0.00 | 50.75 | 0.00 | 0.00 | 48.75 |
| 10:14:14 PM IST | all | 3.00 | 0.00 | 50.75 | 0.25 | 0.00 | 46.00 |
| 10:14:16 PM IST | all | 0.50 | 0.00 | 50.88 | 0.00 | 0.00 | 48.62 |
| 10:14:18 PM IST | all | 0.75 | 0.00 | 50.50 | 0.00 | 0.00 | 48.75 |
| 10:14:20 PM IST | all | 0.50 | 0.00 | 50.87 | 0.25 | 0.00 | 48.38 |
| 10:14:22 PM IST | all | 3.26 | 0.00 | 50.63 | 0.00 | 0.00 | 46.12 |
| 10:14:24 PM IST | all | 0.75 | 0.00 | 51.00 | 0.00 | 0.00 | 48.25 |

**DMA Interrupt:**

| **10:16:44 PM IST** | **CPU** | **%user** | **%nice** | **%system** | **%iowait** | **%steal** | **%idle** |
|---|---|---|---|---|---|---|---|
| **10:16:46 PM IST** | all | 0.25 | 0.00 | 50.25 | 0.00 | 0.00 | 49.50 |
| **10:16:48 PM IST** | all | 3.24 | 0.00 | 49.88 | 0.00 | 0.00 | 46.88 |
| **10:16:50 PM IST** | all | 0.25 | 0.00 | 51.01 | 0.00 | 0.00 | 48.74 |
| **10:16:52 PM IST** | all | 0.50 | 0.00 | 50.87 | 0.00 | 0.00 | 48.63 |
| **10:16:54 PM IST** | all | 3.01 | 0.00 | 50.88 | 0.00 | 0.00 | 46.12 |
| **10:14:22 PM IST** | all | 3.26 | 0.00 | 50.63 | 0.00 | 0.00 | 46.12 |
| **10:14:24 PM IST** | all | 0.75 | 0.00 | 51.00 | 0.00 | 0.00 | 48.25 |

**MMAP:**

| **10:54:39 PM IST** | **CPU** | **%user** | **%nice** | **%system** | **%iowait** | **%steal** | **%idle** |
|---|---|---|---|---|---|---|---|
| **10:54:44 PM IST** | all | 31.66 | 0.00 | 16.08 | 0.00 | 0.00 | 52.26 |
| **10:54:45 PM IST** | all | 36.36 | 0.00 | 17.68 | 0.00 | 0.00 | 45.96 |

**MMAP Interrupt:**

| 11:05:33 PM IST | CPU | %user | %nice | %system | %iowait | %steal | %idle |
|---|---|---|---|---|---|---|---|
| **11:05:39 PM IST** | all | 16.08 | 0.00 | 11.06 | 0.00 | 0.50 | 72.36 |
| **11:05:40 PM IST** | all | 34.00 | 0.00 | 19.00 | 0.00 | 0.00 | 47.00 |
| **11:05:41 PM IST** | all | 29.65 | 0.00 | 15.58 | 0.00 | 0.00 | 54.77 |