CS614:Linux Kernel Programming
Assignment 3 **Name : Nihal Rao**
Date : Apr 20 2023 **Roll no: 22111044 (MTech CSE)**

Design

- Character device (chardev) is used as an interface between user space library and the device driver (cryptocard_mod.c).
- User space library sends input to cryptocard device by writing to the chardev and reads output from cryptocard device by reading from the chardev.
- The chardev open/close/read/write/mmap functions are defined in the device driver.
- Hence functions in the user space library are implemented as a series of write and read calls to the character device.
- For MMIO, encryption/decryption message buffer is stored at address 0xa8 in the device memory.
- For DMA, a coherent DMA buffer of size 4096 bytes is used.

Implementation Details

0.1 Device Driver

- The driver has 2 main logical tasks: to define probe/remove functions to setup and teardown the device and define the chardev interface for communication with the userspace library.
- Probe function does the following:
  - Enables the device using pci_enable_device().
  - Requests MMIO resources using pci_select_bars() and pci_request_region().
  - Maps the MMIO space using ioremap()
  - Register IRQ handler ona shared interrupt line using request_irq()
  - Make the mmio pointer available throughout the driver using pci_set_drvdata().

- Remove function does the following:
  - Free irq line using free_irq().
  - Unmap the MMIO region using iounmap().
  - Disable the device using pci_disable_device().
  - Release the PCI region using pci_release_region().
- Interrupt handler of type irqreturn_t is also defined which does the following:
  - Read the interrupt status register value.
  - If ISR is 0, return IRQ_NONE (interupt was not generated by cryptocard device).
  - Else, write the value of ISR to interrupt ack register, wake up the sleeping process and return IRQ_HANDLED.
- Chardev write function (handles writes to the chardev) does the following:
  - The incoming message from userspace library has a specific format - the first byte of the message is either '0', '1', '2','3','4','5' or '6' corresponding to encryption, decryption, setting keys, setting DMA value, setting INTERRUPT value, encryption using mmaped device or decryption using mmaped device.
  - The remaining part of the incoming message contains data to be encrypted/decrypted or key/config values.
  - For encryption and decryption, the contents of the message are copied to 0xa8 in the device memory region. 0xa8 is then written to data address field (0x80). If interrupts are enabled, wait_event_interruptible() is used to go to sleep until operation is complete.
  - For setting keys, the entire key is first constructed from key a and b, then it is written to 0x08.
  - For setting/unsetting interrupts, the MMIO status register(0x20) is modified accordingly.
- Chardev read function (handles reads to the chardev) does the following:
  - Waits until 0th bit of MMIO status register is cleared (indicates that operation has completed).

  - Copies the encrypted/decryped output from the buffer at 0xa8.
  - Send this output to user using copy_to_user.

- Chardev mmap function is used to mmap the device mmio memory to userspace.

0.2 Userspace Library

- The userspace library defines the create_handle(), close_handle(), encrypt(), decrypt(), set_key() and set_config() functions.
- Encrypt and decrypt functions must take care of the cases when the data length is more than the available buffer size (0xfffff-0xa8). In this case encryption/decryption is done chunk by chunk.
- Userspace library sends messages to the chardev as per the format defined in the chardev's read function: the first byte of the message is either '0', '1', '2','3','4','5' or '6' corresponding to encryption, decryption, setting keys, setting DMA value, setting INTERRUPT value, encryption using mmaped device or decryption using mmaped device.
- create_handle() uses open() call to open the character device and returns the corresponding file descripter.
- close_handle() uses close() to close the file descripter corresponding to chardev.

Test strategies

- The provided test case (test1.c) checks encryption and decryption in MMIO mode without interrupt.
- This file is modified to check various cases such as setting/unsetting interrupts, setting key values, calling encryption/decryption with large data size.
- Logs are also printed from the device driver to dmesg to aid in debugging.

Benchmark results

- The benchmark results are obtained using sar tool.
- All benchmarking tests were run with shared memory file of 10 MB.

	%user	%nice	%system	%iowait	%steal	%idle
mmio	0.81	0.00	51.84	2.37	0.03	44.96
mmio_interrupt	0.08	0.00	46.90	3.68	0.01	49.33
dma	0.03	0.00	50.31	1.96	0.07	47.64
dma_interrupt	0.54	0.00	1.01	1.69	0.01	96.76
mmap	34.36	0.00	1.96	5.03	0.00	58.66
mmap_interrupt	45.66	0.00	4.05	3.47	0.00	54.35

References

- https://olegkutkov.me/2021/01/07/writing-a-pci-device-driver-for-linux/
- Youtube series on pci driver by Johannes 4GNU_Linux
- LDD 3rd edition