4. Linux Character Drivers

Hvad er en character driver?

- Et kerne modul der håndterer skrivning og læsning fra en character device
  - En character device er et device der modtager data i en strøm
  - F.eks. en seriel port, netværks kommunikation, SPI device
  - **Ikke** HDD!
- Bliver tilgået igennem en **device node**, som regel i /dev/

Forklar hvad major-/minor numre er og hvordan de allokeres

- Angiver hvilket kernemodul der skal benyttes
  - En form for adressering af moduler fra nodes
- En character driver har typisk kun et major nummer
  - Minor numre benyttes så til at tilgået "subdevice"
    - F.eks. en ADC med flere kanaler kan have minor no. 0 - N-1, hvor N er antallet af kanaler
- Allokeres ved indsaettelse af kernemodul, og dernæst enten statisk eller dynamisk:
- Statisk:
  - Et modul har et bestemt major no.
  - Ikke egnet til systemer hvor man ikke har styr på hvilke moduler bliver indsat
  - Egnet til små embeddede systemer

Hvordan registrerer man et device og hvad sker der når man gør det?

- Først skal kernemodulet indsættes
- Dernæst skal der oprettes et device node

```
mknod /dev/mychardev -c 62 0
```

- Der laves et node i /dev/ med navnet mychardev, der angives det er en char device med major no. 62, og minor no. 0
- Herefter kan device driveren tilgås fra userspace applikationer som en fil

Forklar formålet med metoderne Open / Close / Read / Write

- Metoderne benyttes til at bestemme hvad der sker, når man benytter forskellige kaldt på character devicets device node
  - Når noden åbnes, vil open metoden blive kaldt
  - Når der skrives eller læses fra noden bliver read/write kaldt
  - Når noden lukkes, vil close metoden blive kaldt

- Disse metoder kaldes **Fil Operationer**, altså de operationer der udføres når devicet benyttes som en fil
  - Disse operationer skal erklæres i **FOPS Structen**, således driveren ved hvad der skal kaldes, og hvornår

Forklar hvordan data overføres mellem user- / kernel space

- ```
  unsigned long copy_to_user(void __user *to, const void *from, unsigned long n);
  ```
  - Copies a block of data, with a specified size, to userspace
  - 'from' is the source address in kernelspace
  - 'to' is the destination address in userspace
  - 'n' is the size of the block in bytes
  - returns the amount of bytes that could **not** be copied. 0 on success
- read og write metoder tager en char* ind som argument, som kan benytes til at kopiere til og fra userspace
- Typisk i read metoden, vil der blive kopieret data til userspace
- I write vil der typisk blive kopieret fra userspace
  ```