Entwurfsdokument
BS Praktikum 4
Swaneet Sahoo und Ivan Morozov
7.1.14

Entwurf des Moduls translate:

Das Translate-Modul wurde ähnlich dem scull-Beispiel entwickelt. Im Header wird das struct translate_dev definiert, welches diverse Pointer für die Buffer sowie das struct cdev und zwei Semaphoren für den Lese-/Schreibzugriff hat. Im Header werden außer den Funktionsdeklarationen auch die Device-Fileoperations implementiert(bzw. wird dort auf die einzelnen Funktionen verwiesen.) Im Header stehen außerden noch globale Konstanten und hilfreiche Makros die z.B. für die Kodierung/Dekodierung verwenden werde.

Das translate_open versucht je nach Modus auf dem dazugehörigen Semaphoren zu blockieren und sendet entweder einen Erfolg oder ein EBUSY zurück.
Das Translate_close gibt einfach den entsprechenden Semaphroren frei.
Im translate_write werden nach und nach die Chars aus dem Userspace kopiert. Falls es sich um das translate0 device handelt, wird es außerdem noch verschlüsselt. Ist zwischendurch der Buffer voll, wird vorzeitig abgebrochen.
Im translate_read passiert das genaue Gegenteil. Es wird dort Char für Char ins Userspace kopiert und im Falle der translate1 dekodiert. Ist zwischendurch der Buffer leer, wird vorzeitig abgebrochen.

Bei der Kodierung wird anhand des ASCII aus dem translate_substr gelesen durch welchen Buchstaben es zu ersetzten ist. Die ersten n/2 Buchstben geben die Kodierung für Kleinbuchstaben an. (n steht für die Länge des translate_substr). Die anderen Buchstaben geben die Kodierung für Großbuchstaben an.
Für das Dekodieren wird aus die mit strchar() berechnete Position des Chars im im translate_substr die ASCII Zahl des kodierten Buchstabens berechnet. Damit erhält man dann das Original wieder.

Das translate_cleanup, translate_init und translate_setup_cdev wurden aus scull übernommen. Dort werden die Werte des translate_dev struct initialisiert/zurückgesetzt und der Speicher für die Devices und die Buffer werden alloziert oder freigegeben.

Das Makefile ist fast identisch mit dem aus dem scull. Zwei Scripte install.sh und uninstall.sh übernehmen das (De-)Installieren des Moduls. Das Install.sh macht das gleiche wie der Script für die scull-Installation aus dem dritten Kapitel aber führt davor noch die Kompilierung aus. Das uninstall.sh ruft einfach ein rmmod auf, entfernt die Device-Nodes und löscht alle Kompilationsdateien.