4 Ein Kernelmodul mit Stoppuhrfunktion

Schreiben Sie einen Gerätetreiber für zwei Geräte mit den Device-Nodes `/dev/timerf` und `/dev/timerr`, die jeweils einen vorwärts- und einen rückwärtszählenden Timer realisieren.

- Das meiste, was Sie zur Bearbeitung der Aufgabe wissen müssen steht in den Kapiteln 1 – 5 des Buches
  „Linux Device Drivers“ von Jonathan Corbet, Alessandro Rubini, und Greg Kroah-Hartman, die URL des ersten Kapitels ist `http://oreilly.com/catalog/linuxdrive3/book/ch01.pdf`.
- Die zugehörigen Beispielprogramme finden Sie unter der URL `http://examples.oreilly.com/linuxdrive3/examples.tar.gz`. Ich habe die Programme aus dem Ordner `scull` so modifiziert, dass sie sich auf unseren virtuellen Maschinen kompilieren lassen und auf meinem Pub-Verzeichnis abgelegt:
  `https://pub.informatik.haw-hamburg.de/home/pub/prof/fohl/Bs/Praktikum/ldd_examples_scull_2015.tar.bz2` Schauen Sie sich insbesondere das Programm `pipe.c` gründlich an. Dort finden Sie alles, was Sie für die Aufgabe brauchen
- Hinweise zum Umgang mit Kernel-Timern finden Sie hier:
  `http://www.linux-magazin.de/Ausgaben/2006/04/Kern-Technik`

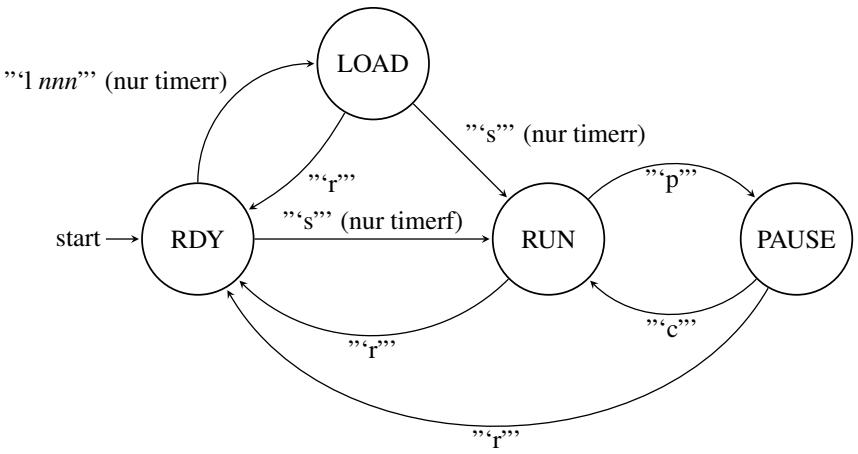
4. 1 Funktion

Das Stoppuhr-Device funktioniert so, dass durch Schreiben von Befehlscodes die Stoppuhr ihren Zustand ändert, und durch Lesen der aktuelle Zeitwert in *Jiffies* ausgegeben wird. Jiffies sind die internen Timer-Ticks des Kernels und werden mit der Funktion

```
#include <linux/jiffies.h>
....
  u64 get_jiffies_64()
```

ermittelt.

4. 1.1 Befehlscodes und Zustandsübergänge



Start Zustand = READY

s Timer starten Vorwärtszähler: READY -> RUNNING
Rückwärtszähler: LOADED -> RUNNING

p Pause: RUNNING -> PAUSE

c Continue: PAUSE -> RUNNING

r Reset: <ALL> -> READY
alle internen Zeitwerte löschen

l <value> Laden (nur für Rückwärtszähler): READY -> LOADED

4. 1.2 Ausgabe

Beim Lesen von Ihrem Device soll der Vorwärtszähler `/dev/timerf` die Jiffies seit dem Start abzüglich der Pausenzeiten ausgeben.

Der Rückwärtszähler zählt vom geladenen Wert herunter, solange er nicht pausiert ist. Wenn die aktuelle Zeitdifferenz kleiner als die geladene Zeit ist, wird Null ausgegeben.

Hinweise:

- Vorsicht bei der Differenzbildung von **`unsigned`**-Werten: Sie können nicht auf ein negatives Resultat testen!
- Sämtliche Zeitberechnungen finden nur in der `read`- Methode statt. Sie müssen nicht dafür sorgen, dass Ihr Modul im Hintergrund ständig die Timer-Ticks aufsummiert.

4. 1.3 Tipps

Für die hier gezeigten Verfahren zur Auswertung des User-Inputs brauchen Sie das Header-File `linux/string.h`.

Auswerten der Befehls-Zeichen Nach `copy_from_user` mögen die Daten auf der String-Variablen `instring` liegen, dann erhalten Sie mit

```
cmdchar = strchr(instring, 's');
```

Einen Zeiger auf das 's', falls vorhanden, sonst einen Null-Zeiger.

Umwandeln der Zahl für die zu ladende Zeit Nachdem `cmdchar` auf den Befehlscode 'l' zeigt, kann mit

```
int success;
unsigned long timeval;
...
success = sscanf(cmdchar+1, "%lu", &timeval);
```

der Rest des Input-Strings in eine unsigned long-Zahl gewandelt und in `timeval` gespeichert werden. Sie müssen auf `success == 1` prüfen, sonst steht auf `timeval` Murks.

4. 2 Aufgabe

- Schreiben Sie ein Kernelmodul, das die geforderte Funktionalität realisiert.
- Das Modul *muss* aus *einer* C-Datei (und natürlich einer Header-Datei) bestehen.
- Das Modul *muss* `timer` heißen, das Kernel-Objectfile *muss* `timer.ko` heißen
- Das Modul *muss* seine Major-Nummer dynamisch vom Kernel erhalten.

Mit anderen Worten: Sie müssen die benötigte Funktionalität aus den Quellcodefiles `main.c` und `pipe.c` zusammenklauben und in eine Datei stopfen, und alle Erinnerungen an *scull* und *pipe* beseitigen. Außerdem müssen Sie das Makefile anpassen.

4. 3 Installation

Zur Insatallation des Moduls schreiben Sie zur Schonung Ihrer eigenen Nerven ein Skript, das folgendes tut:

- Alte Device-Nodes `/dev/timer?` entfernen.
- Altes Kernel-Modul entfernen.
- Neues Kernel-Modul laden.
- Major-Devicenummer aus `/proc/devices` mit `grep` und `cut` ermitteln.
- Neue Device-Nodes `/dev/timerf` und `/dev/timerr` mit der korrekten Major-Nummer erstellen.

4. 4 Hinweise

- Gehen Sie großzügig mit Debug-Meldungen per `PDEBUG` um.

4. 5 Test / Abnahme

Folgende Dinge werde ich bei der Abnahme testen:

- Automatische Installation von Kernelmodul und Device-Nodes.
- Beobachten der Debug-Meldungen im Kernel-Log.
- Timer-Funktionalität mit mehrfachen Pausen, ungültigen Befehlscodes und Load-Werten usw.