Ninjastorms Team Kernel

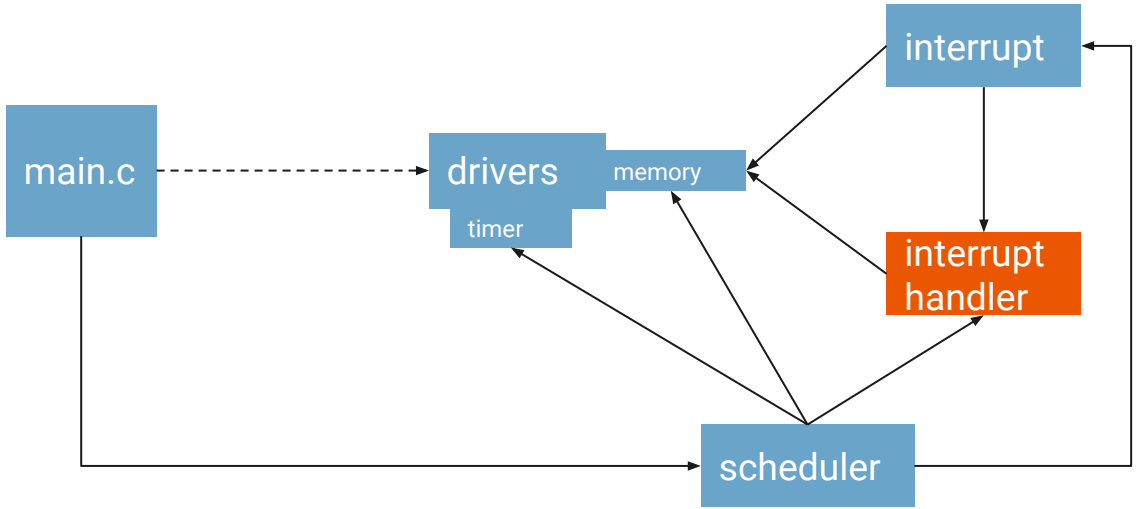
Felix Roth und Konrad Hanff

https://github.com/hpi-bs2-st2020-ninjastorms-kernel/ninjastorms

Ausgangssituation

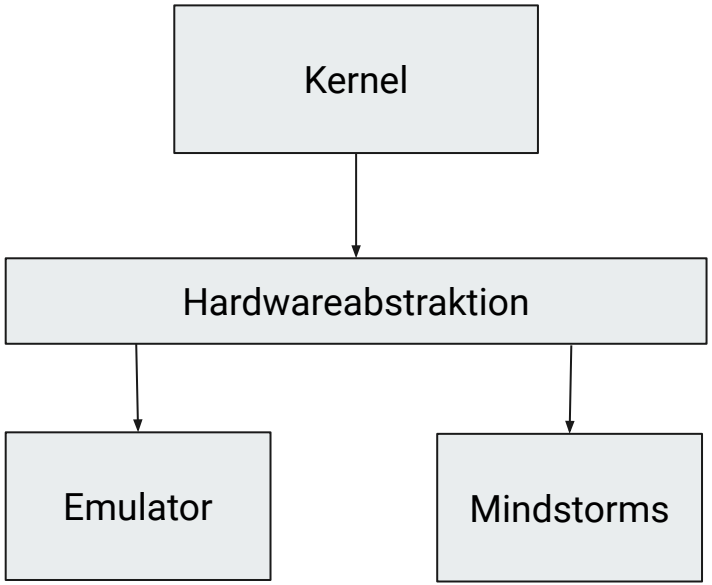
- System läuft im Emulator ohne nennenswerte Probleme
- Printen von einzelnen Zeilen auf Konsole möglich
- spezielle C Library für das Betriebssystem
- Scheduler für Tasks mit Ringbuffer (inkl. Timerinterrupts)
- Mindstorms-Geräte ansteuern

Struktur


interrupt
main.c
drivers
memory
timer
interrupt handler
scheduler

Unsere Ziele

Hardwareabstraktion



Interface für Timer, Speicher (Memory-mapped registers), Geräte, ...

tatsächliche Speicheradressen

Interrupts

- Grundlagen sind gelegt (Dispatcher, Trap, IRQL Einteilung)
- Wo werden ISRs vermerkt?
- Dispatcher und ISR trennen
- Vorraussetzung für Netzwerkkommunikation, Syscalls, Paging, ...

Prozesse und Scheduler

- Interrupt Handler und Scheduler sind momentan noch etwas vermischt
- aktuelle “Task”-Struktur zu einem Prozesskonzept mit PID und z.B. *eigenem Speicherbereich* erweitern
- Scheduler an neue Prozesse anpassen
- Wie entstehen Prozesse?

- eventuell Child-Prozesse (nur ganz vielleicht)

Syscalls

- Trennung User Mode / Kernel Mode muss aufgebaut werden
- User Mode library
- Prozesse im User Mode
- basiert auf funktionierender Interrupt-Behandlung
- benötigt ISR, die nach Registerinhalt spezielle Aufgaben übernimmt
- Bsp: Prozess erstellen, *Speicher allozieren*, Interprozesskommunikation, ...

Weitere Herausforderungen und Schritte

- Koordination mit 2 anderen Gruppen am selben Projekt
  - hoffentlich können die Projekte am Ende leicht zusammengeführt werden
- Nutzerinteraktion am Terminal
- C Standardbibliothek weiter implementieren
- …