Übungen zu Systemnahe Programmierung in C (SPiC) – Sommersemester 2019

Übung 7

Benedict Herzog
Bernhard Heinloth

Lehrstuhl für Informatik 4
Friedrich-Alexander-Universität Erlangen-Nürnberg

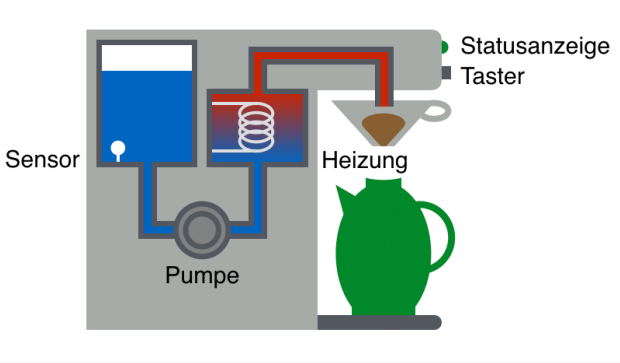
Lehrstuhl für Verteilte Systeme und Betriebssysteme

FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Vorstellung Aufgabe 4

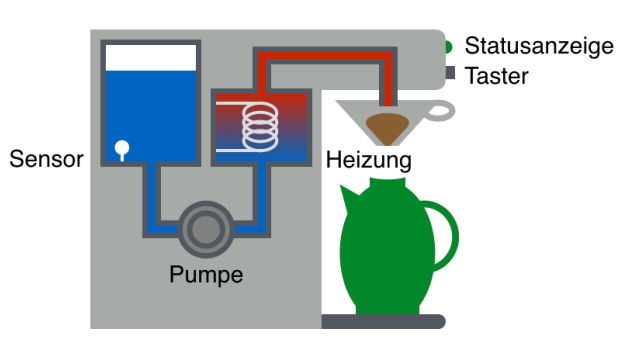
Hands-on: Kaffeemaschine

Hands-on: Kaffeemaschine (1)



- Lernziele:
  - Zustandsmaschine
  - Interrupts & Schlafenlegen
  - Timer bzw. Alarm

Hands-on: Kaffeemaschine (1)



- Beschaltung:
  - Pumpe & Heizung: Port D, Pin 5 (active-low)
  - Taster: `INT0` an Port D, Pin 2 (active-low)
  - Sensor: `INT1` an Port D, Pin 3 (Wasser: high; kein Wasser: low)
  - Statusanzeige:
    - `BLUE0`: **STANDBY**
    - `GREEN0`: **ACTIVE**
    - `RED0`: **NO_WATER**

Hands-on: Kaffeemaschine (2)

STANDBY

- Kaffeemaschine ist aus
- Pumpe und Heizung aus
- Startmodus
- Benutzer kann Kaffeezubereitung durch Tastendruck starten

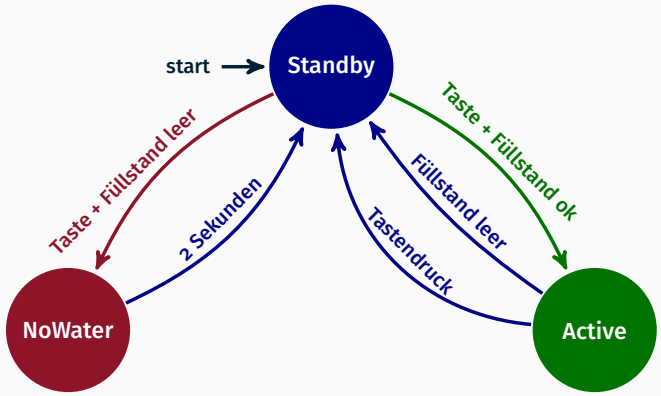
ACTIVE

- Kaffeemaschine ist an
- Pumpe und Heizung sind an
- Wassertank ist nicht leer
- Benutzer kann Kaffeezubereitung durch Tastendruck beenden

NO_WATER

- Kaffeemaschine zeigt an, dass sie nicht genügend Wasser hat
- Zeitdauer: 2 Sekunden

Hands-on: Kaffeemaschine (2)



- Hinweise:
  - Tastendruck & Füllstandsänderung durch Interrupts
  - Statusanzeige: `void setLEDState(km_state state)`
  - Wartephasen ggf. über Singleshot-Alarm realisieren
  - In Wartephasen Mikrocontroller in den Energiesparmodus

Hands-on: Kaffeemaschine (3)

DDRx hier konfiguriert man Pin i von Port x als Ein- oder Ausgang
- `Bit i = 1` → Pin i als Ausgang verwenden
- `Bit i = 0` → Pin i als Eingang verwenden

PORTx Auswirkung **abhängig von DDRx**:
- ist Pin i **als Ausgang konfiguriert**, so steuert Bit i im PORTx Register ob am Pin i ein high- ode r ein low-Pegel erzeugt werden soll
  - `Bit i = 1` → high-Pegel an Pin i
  - `Bit i = 0` → low-Pegel an Pin i
- ist Pin i **als Eingang konfiguriert**, so kann man einen internen pull-up-Widerstand aktivieren
  - `Bit i = 1` → pull-up-Widerstand an Pin i (Pegel wird auf high gezogen)
  - `Bit i = 0` → Pin i als tri-state konfiguriert

PINx Bit i gibt aktuellen Wert des Pin i von Port x an (nur lesbar)

Hands-on: Kaffeemaschine (4)

- Interrupt Sense Control (ISC) Bits befinden sich beim ATmega328PB im External Interrupt Control Register A (EICRA)
- Position der ISC-Bits im Register durch Makros definiert

Interrupt 0		Interrupt bei	Interrupt 1	
ISC01	ISC00		ISC11	ISC10
0	0	low Pegel	0	0
0	1	beliebiger Flanke	0	1
1	0	fallender Flanke	1	0
1	1	steigender Flanke	1	1

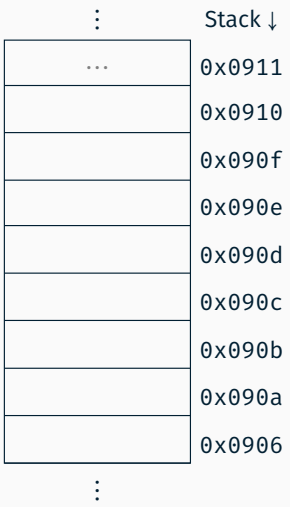
- ATmega328PB: External Interrupt Mask Register (EIMSK)
- Die Bitpositionen in diesem Register sind durch Makros INTn definiert

Hands-on: Laufschrift

Vertiefung: Strings

- char: Einzelnes Zeichen (z.B. 'a')
- String: Array von chars (z.B. "Hello")
- In C: Letztes Zeichen eines Strings: '\0'
  ⇒ Speicherbedarf: strlen(s) + 1

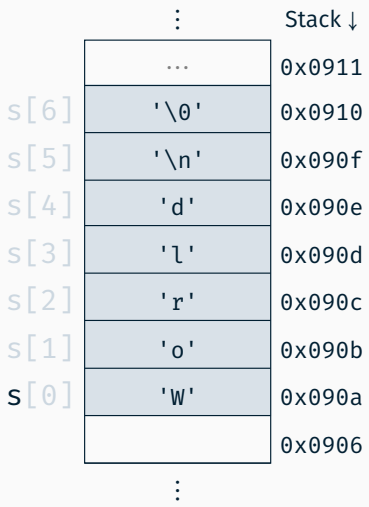
```
01 char *s = "World\n";
```



Vertiefung: Strings

- char: Einzelnes Zeichen (z.B. 'a')
- String: Array von chars (z.B. "Hello")
- In C: Letztes Zeichen eines Strings: '\0'
  ⇒ Speicherbedarf: strlen(s) + 1

```
01 char *s = "World\n";
```



Hands-on: Laufschrift

- Funktionsweise:
  Schrittweises Anzeigen eines Textes auf der 7-Segment-Anzeige
- Lernziele:
  - Alarme & Schlafenlegen
  - Zeiger & Zeigerarithmetik
  - Zeichenfolgen in C
- Vorgehen:
  - Aktivieren eines wiederkehrenden Alarms (`sb_timer_setAlarm()`)
  - Optional: Manuelles Konfigurieren des Timers
  - Zusammensetzen des aktuellen Teilstrings
  - Ausgabe über 7-Segment-Anzeige
  - In Wartephasen Mikrocontroller in den Energiesparmodus versetzen