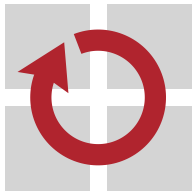
Übungen zu Systemnahe Programmierung in C

Abschnitt 3.1: Variablen

11.05.2020

Tim Rheinfels
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

Verwendung von `int`

- Die Größe von `int` ist nicht genau definiert
- Zum Beispiel beim ATmega328PB: 16 bit
  - ⇒ Gerade auf µC führt dies zu langsamerem Code und/oder Fehlern
- Für die Übung gilt
  - Verwendung von `int` ist ein Fehler
  - Stattdessen: Verwendung der in der `stdint.h` definierten Typen: `int8_t`, `uint8_t`, `int16_t`, `uint16_t`, etc.
- Wertebereich
  - `limits.h`: `INT8_MAX`, `INT8_MIN`, ...
- Speicherplatz ist auf µC sehr teuer
  (SPiCboard/ATmega328PB hat nur 2048 Byte SRAM)

⇝ Nur so viel Speicher verwenden, wie tatsächlich benötigt wird!

Typedefs & Enums

```
01 #define PB3 3
02
03 typedef enum {
04     BUTTON0 = 0, BUTTON1 = 1
05 } BUTTON;
06
07 typedef enum {
08     PRESSED = 0, RELEASED = 1, UNKNOWN = 2
09 } BUTTONSTATE;
10
11 void main(void) {
12     /* ... */
13     PORTB |= (1 << PB3); // nicht (1 << 3)
14
15     // Deklaration: BUTTONSTATE sb_button_getState(BUTTON btn);
16     BUTTONSTATE zustand = sb_button_getState(BUTTON0); // nicht
        ↪ sb_button_getState(0)
17     /* ... */
18 }
```

- Vordefinierte Typen verwenden
- Explizite Zahlenwerte nur verwenden, wenn notwendig