

Übungen zu Systemnahe Programmierung in C (SPiC)

Peter Wägemann, Sebastian Maier, Heiko Janker
(Lehrstuhl Informatik 4)



Sommersemester 2015



Bitoperationen

Shiftoperationen

Aufgabe 3: Geschicklichkeitsspiel

Hands-on: Binärzähler



- Übersicht:

&	0	1
0	0	0
1	0	1

	0	1
0	0	1
1	1	1

^	0	1
0	0	1
1	1	0

~	
0	1
1	0

- Beispiel:

	1100	1100	1100
~	&		^
1001	1001	1001	1001
0110	1000	1101	0101



- Beispiel:

```
uint8_t x = 0x9d; 

|   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 |
|---|---|---|---|---|---|---|---|

  
x <<= 2; 

|   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 |
|---|---|---|---|---|---|---|---|

  
x >>= 2; 

|   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 1 | 1 | 0 | 1 |
|---|---|---|---|---|---|---|---|


```

- Setzen von Bits:

```
(1 << 0) 

|   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
|---|---|---|---|---|---|---|---|

  
(1 << 3) 

|   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
|---|---|---|---|---|---|---|---|

  
(1 << 3) | (1 << 0) 

|   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 |
|---|---|---|---|---|---|---|---|


```

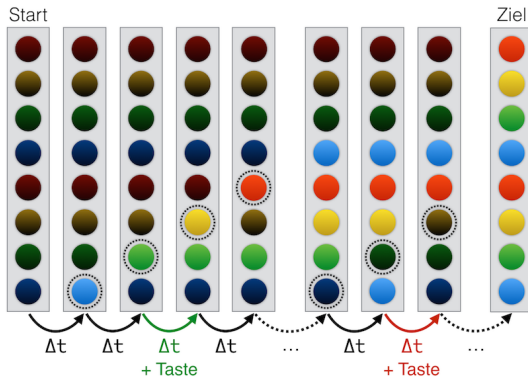
- **Achtung:**

Bei signed-Variablen ist das Verhalten des >>-Operators nicht 100% definiert. Im Normalfall(!) werden bei negativen Werten 1er geshiftet.



Aufgabe 3: Geschicklichkeitsspiel (1)

- Spielcursor wandert dabei über LED-Reihe hin und her und invertiert (engl. toggle) den LED-Zustand
- LED-Zustand bleibt durch Drücken des Tasters erhalten
- Ziel: alle LEDs zum Leuchten bringen



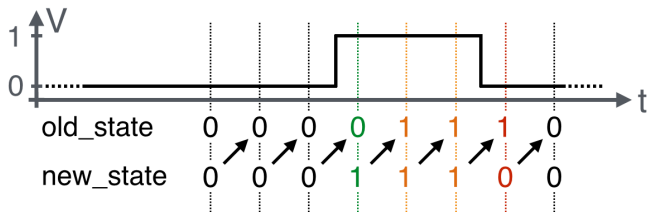
Aufgabe 3: Geschicklichkeitsspiel (2)

- Schwierigkeit (Geschwindigkeit) steigt mit jedem Level
- Nach einem Level wird eine Siegessequenz auf den LEDs dargestellt

```
1 static void play(uint8_t level)
2 static void show_win(void);
3
4 void main(void) {
5     uint8_t level = 1;
6
7     while(1){
8
9         play(level);
10
11        show_win();
12
13        // Level aktualisieren
14    }
15
16 }
```



Aufgabe 3: Flankendetektion ohne Interrupts



- Detektion der Flanke durch aktives, **zyklisches Abfragen** (engl. Polling) eines Pegels
- Später: Realisierung durch Interrupts



- Zähler wird durch Button 0 inkrementiert
- Zähler kann durch Button 1 zurückgesetzt werden
- Anzeige erfolgt via 7-Segment Anzeige und binär auf den LEDs
- Betätigung der Tasten soll durch Flankendetektion erkannt werden
- Optional:
 - Anzeige der Programmversion bei Programmstart (z.B. "v2")
 - Umkehrung der LED Reihenfolge
 - ...

