

Übungen zu Systemnahe Programmierung in C (SPiC) – Sommersemester 2018

Übung 3

Benedict Herzog
Sebastian Maier

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



Lehrstuhl für Verteilte Systeme
und Betriebssysteme



FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

TECHNISCHE FAKULTÄT

Bits & Bytes



■ 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

1



■ 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

1

■ 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:

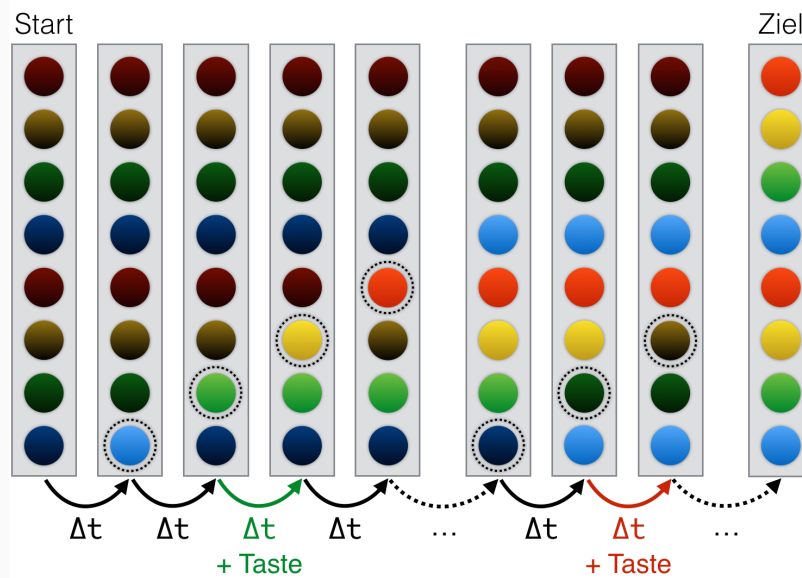
(1 << 0)	0	0	0	0	0	0	0	1
(1 << 3)	0	0	0	0	1	0	0	0
(1 << 3) (1 << 0)	0	0	0	0	1	0	0	1

■ Achtung:

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

Aufgabe: Geschicklichkeitsspiel

- 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



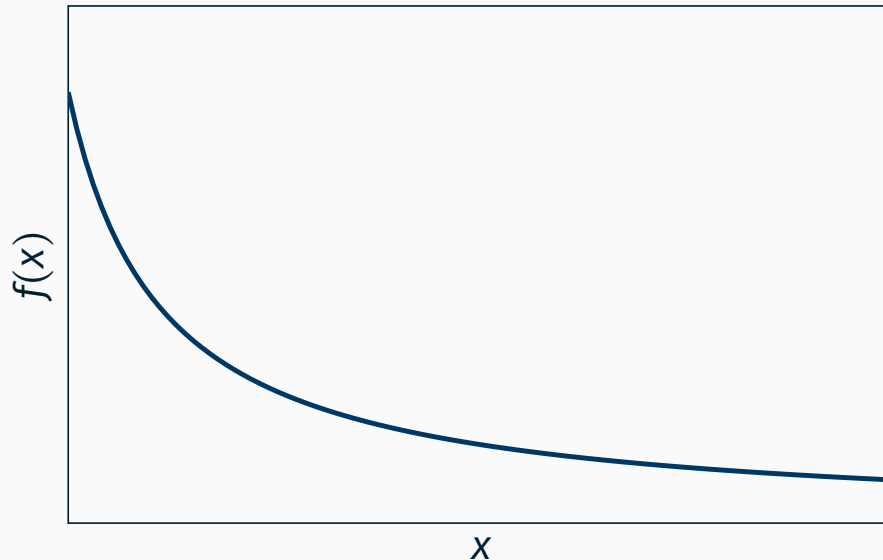
3

- Nach einem Level wird eine Siegessequenz auf den LEDs dargestellt

```
01 static void play(uint8_t level)
02 static void show_win(void);
03
04 void main(void) {
05     uint8_t level = 1;
06
07     while(1){
08
09         play(level);
10
11         show_win();
12
13         // Level aktualisieren
14     }
15
16 }
```

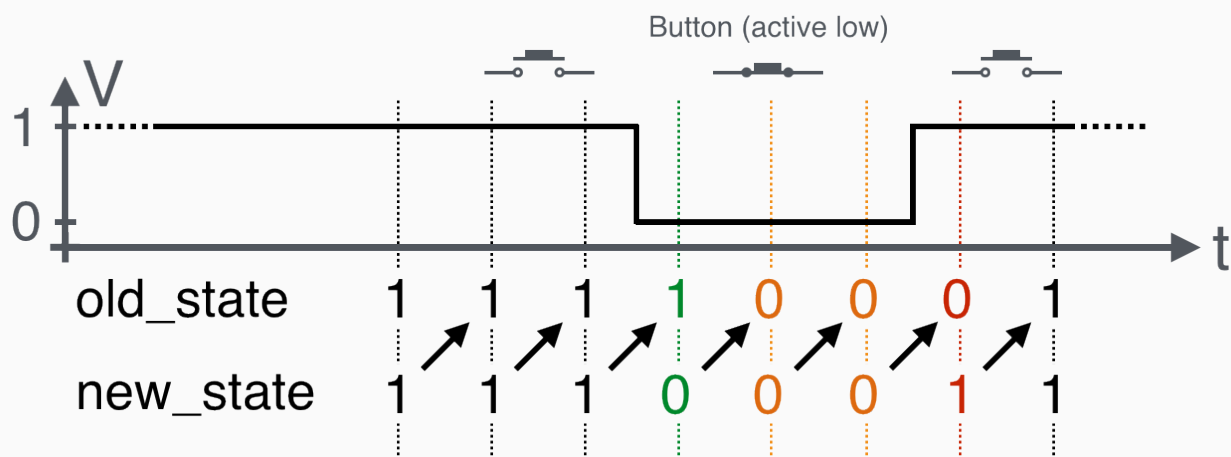
4

- Schwierigkeit (Geschwindigkeit) steigt mit jedem Level
- Schwierigkeit nähert sich einer maximalen Geschwindigkeit
- Asymptotische Annäherung: $f(x) = \frac{c}{x}$ (c ist eine Konstante)



5

Aufgabe: Flankendetektion ohne Interrupts



- Detektion der Flanke durch aktives, **zyklisches Abfragen** (engl. Polling) eines Pegels
- Unterscheidung zwischen **active-high** & **active-low** notwendig
- Später: Realisierung durch Interrupts

6

Hands-on: Binärzähler

Hands-on: Binärzähler



- Ausgabe des aktuellen Zählerstandes
 - binär auf LEDs 0 - 7
 - hexadezimal auf 7-Segment Anzeigen
- Zählvariable durch Tastendruck inkrementierbar
 - Button 0 inkrementiert erste Hexadezimalstelle
 - Button 1 inkrementiert zweite Hexadezimalstelle
- Betätigung der Tasten soll durch Flankendetektion erkannt werden
- *Optional:* `led_setMaskInverted( )` mit invertierter LED-Reihenfolge

