Hochschule Aalen

Aufgabe 5

- Es soll eine FSM für den Komfortblinker eines Autos entwickelt werden.
    - 3 mal Blinken des entsprechenden Blinklichts beim Antippen, bzw. kurzzeitigen Betätigen des Tasters Links bzw. Rechts
    - Dauerhaftes Blinken, solange der Blinkhebelschalter betätigt ist
        - d.h. das Blinklicht blinkt im Normalfall mindestens drei Mal.
    - Beim Wechsel der Seiten soll das Blinken auf der vorherigen Seite beendet werden.
        - In diesem Fall natürlich auch weniger wie 3 mal Blinken möglich!
- Hardware:
    - LED3 und LED2 - linkes Blinklicht
    - LED1 und LED0 - rechtes Blinklicht
    - Taster T3 - Blinkhebel links
    - Taster T0 - Blinkhebel rechts

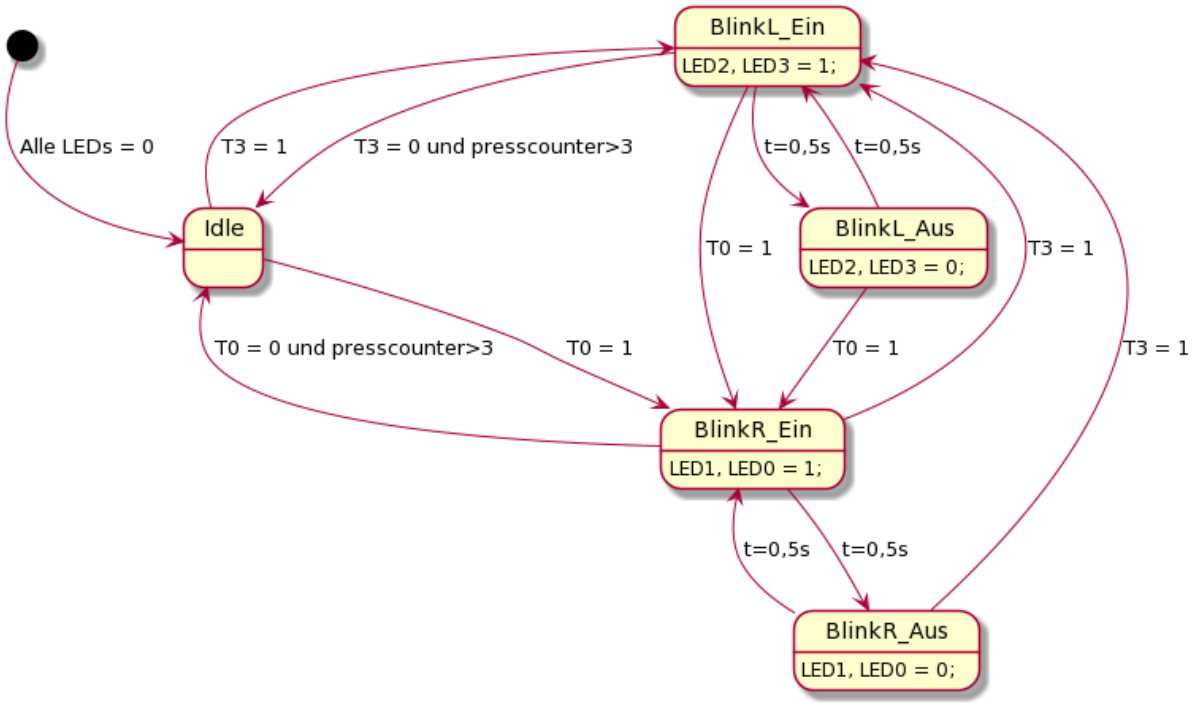
FSM für Komfort-Blinker eines Autos

```
@startuml
[*] -d-> Idle : Alle LEDs = 0
    Idle -right-> BlinkR_Ein : T0 = 1
    Idle -left-> BlinkL_Ein : T3 = 1
    BlinkR_Ein --> Idle : T0 = 0 und presscounter>3
    BlinkL_Ein --> Idle : T3 = 0 und presscounter>3
    BlinkR_Aus --> BlinkL_Ein : T3 = 1
    BlinkL_Aus --> BlinkR_Ein : T0 = 1
        BlinkR_Ein --> BlinkL_Ein : T3 = 1
    BlinkL_Ein --> BlinkR_Ein : T0 = 1


    BlinkR_Ein --> BlinkR_Aus : t=0,5s
    BlinkR_Aus --> BlinkR_Ein : t=0,5s
    BlinkL_Ein --> BlinkL_Aus : t=0,5s
    BlinkL_Aus --> BlinkL_Ein : t=0,5s


    BlinkR_Ein : LED1, LED0 = 1;
    BlinkR_Aus : LED1, LED0 = 0;
    BlinkL_Ein : LED2, LED3 = 1;
    BlinkL_Aus : LED2, LED3 = 0;
@enduml
```

In PlantUML (grafisch):



Nichtblockierende Programmierung des Komfort-Blinkers

Für jeden Zustand wird eine Statefunktion definiert:

```
1.  void *FSM1_Idle(void);
2.  void *FSM1_BlinkR_An(void);
3.  void *FSM1_BlinkR_Aus(void);
4.  void *FSM1_BlinkL_An(void);
5.  void *FSM1_BlinkL_Aus(void);
```

Die Funktionen geben jeweils als return-Wert die Adresse der als nächstes auszuführenden Funktion zurück.

```
1.  void *FSM1_Idle(void)
2.  {
3.      LED0 = LED1 = LED2 = LED3 = 0;
4.      if (T3 == 1)
5.        return FSM1_BlinkL_An;
6.      else if (T0 == 1)
7.        return FSM1_BlinkR_An;
8.      return FSM1_Idle;
9.  }
```

Im Idle-Modus sind alle LEDs aus. Sobald einer der Taster gedrückt ist, wird im nächsten Funktionsaufruf der KomfortBlink_FSM-Funktion eine der beiden Teil-Funktionen (FSM1_BlinkR_An oder FSM1_BlinkL_An) ausgeführt.

```
1.  void KomfortBlink_FSM1(void)
2.  {
3.       static StateFunc statefunc = FSM1_Idle;
4.
5.       statefunc = (StateFunc)(*statefunc)();
6.  }
```

```
1.  void *FSM1_BlinkL_An(void){
2.      static uint16_t counter = 0;
3.      counter++;
4.      if (T0 == 1){
5.          counter = 0;
6.          blinkvorgaengeL_AN = 0;
7.          blinkvorgaengeL_AUS = 0;
8.          LED3 = LED2 = 0;
9.          return FSM1_BlinkR_An;
10.     }
11.     if ((T3 == 1) || blinkvorgaengeL_AN<3){
12.         LED3 = LED2 = 1;
13.         }
14.     else{
15.         if (blinkvorgaengeL_AN > 3){
16.             counter = 0;
17.             blinkvorgaengeL_AN = 0;
18.             blinkvorgaengeL_AUS = 0;
19.             return FSM1_Idle;
20.         }
21.
22.     }
23.     if (counter >= BLINKZEIT){
24.         blinkvorgaengeL_AN++;
25.         counter = 0;
26.         return FSM1_BlinkL_Aus;
27.         }
28.     return FSM1_BlinkL_An;
29.
30. }
```

Zunächst wird abgefragt, ob der Taster für die jeweils andere Seite gedrückt wurde. In diesem Fall werden die LEDs abgeschaltet und es wird in den anderen Zustand gewechselt. Die LEDs werden eingeschaltet, falls der Taster3 gedrückt gehalten wird, oder noch nicht alle 3 Blinkvorgänge für die Komfortblinkfunktion ausgeführt worden sind.

Falls alle 3 Blinkvorgänge abgeschlossen worden sind und der Taster nicht gedrückt gehalten wird wechselt die Machine in den Idle-Modus.

Mit Hilfe eines Counters wird ermittelt, wie oft die Funktion schon aufgerufen worden ist. Sobald der Counter den gewünschten Wert erreicht hat (entspricht hier 500ms), wechselt die Statemachine in den Zustand FSM1_BlinkL_Aus. Außerdem wird mit der Variable blinkvorgaengeL_AN mitgezählt, wie viele Blinkvorgänge schon ausgeführt worden sind.

```
1.  void *FSM1_BlinkL_Aus(void){
2.      static uint16_t counter = 0;
3.      counter++;
4.      if (T0 == 1){
5.          counter = 0;
6.          blinkvorgaengeL_AUS = 0;
7.          blinkvorgaengeL_AN = 0;
8.          LED3 = LED2 = 0;
9.          return FSM1_BlinkR_An;
10.     }
11.     LED3 = LED2 = 0;
12.     if (T3 == 0){
13.         if (blinkvorgaengeL_AUS > 3){
14.             counter = 0;
15.             blinkvorgaengeL_AUS = 0;
16.             blinkvorgaengeL_AN = 0;
17.             return FSM1_Idle;
18.         }
19.     }
20.     if (counter >= BLINKZEIT)
21.     {
22.         blinkvorgaengeL_AUS++;
23.         counter = 0;
24.         return FSM1_BlinkL_An;
25.     }
26.     return FSM1_BlinkL_Aus;
27.
28. }
```

Im FSM1_BlinkL_Aus – Zustand wird nun auch wieder abgefragt, ob der Taster für die rechte Seite gedrückt worden ist . Falls dies der Fall ist, wird in diesen Zustand gewechselt. Die LEDs werden ausgeschaltet. Auch hier sorgt ein Counter dafür, dass die LEDs für eine gewisse Zeit ausgeschaltet bleiben. Sobald der Taster nicht mehr gedrückt wird und die 3 Blinkvorgänge ausgeführt worden sind wechselt die Maschine in den Idle-Modus.

Analog dazu wurden auch die Funktionen für die andere Seite programmiert.