

Interne Ereignisse im Zustandsdiagramm

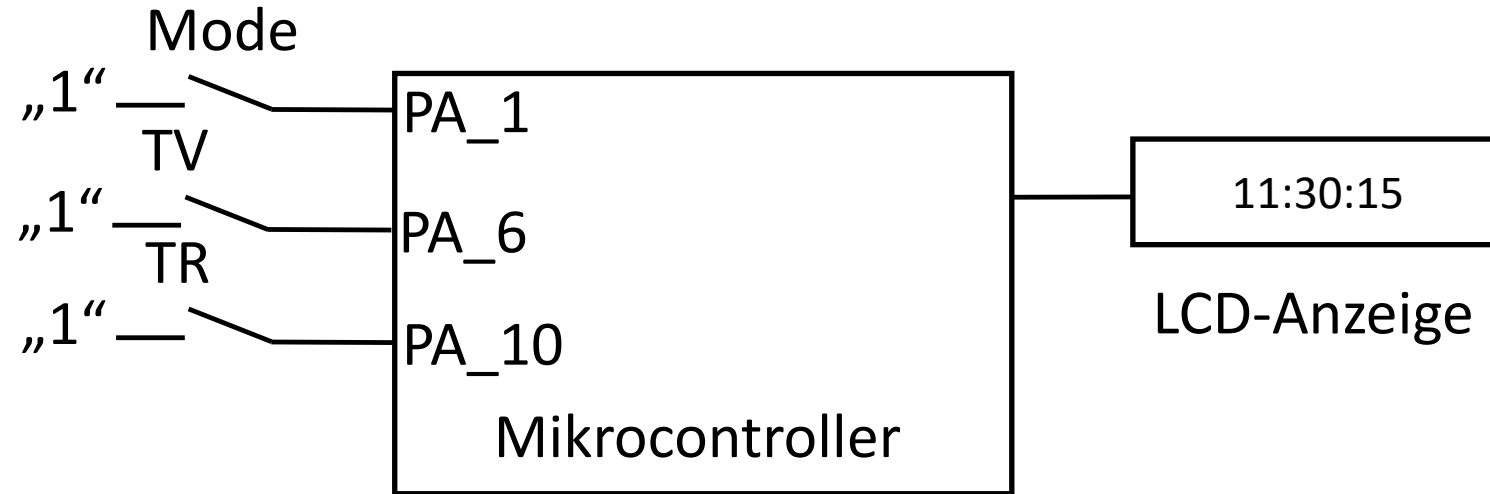
Interrupts



Ich bin Mik, Dein Mikrocontroller



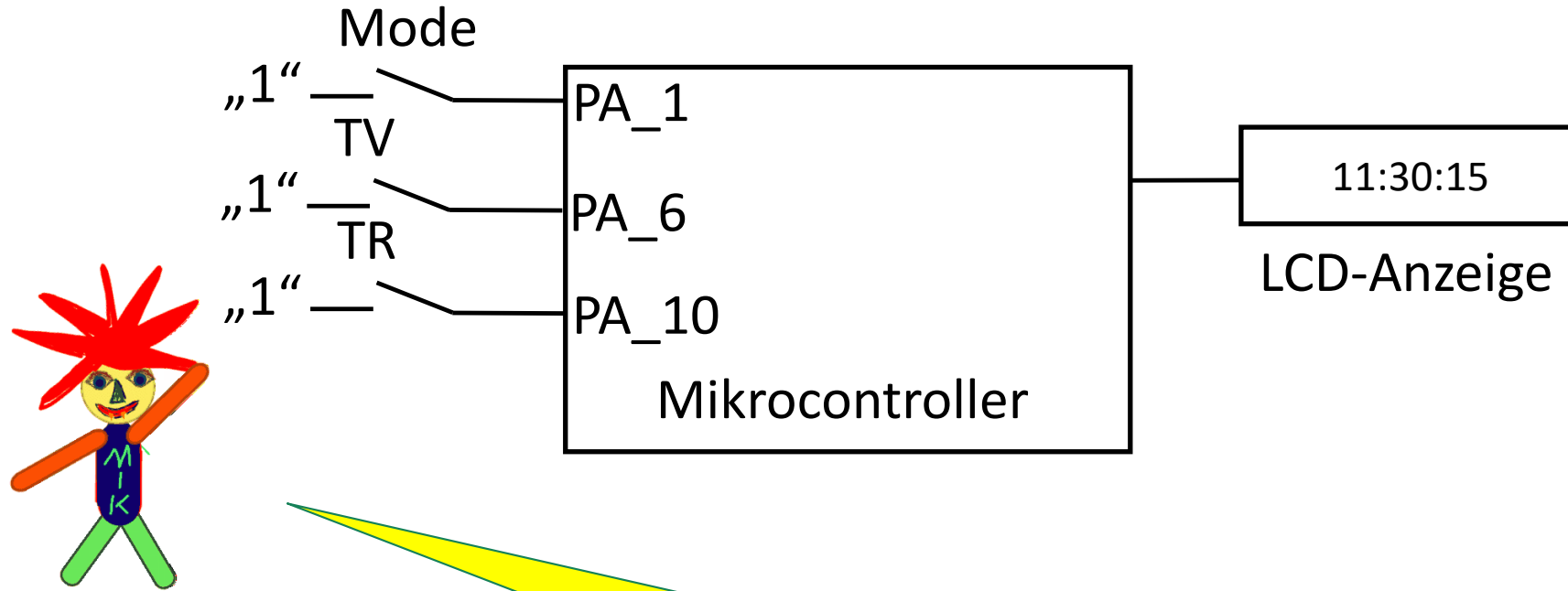
Ereignisse im Zustandsdiagramm



Eine Digitaluhr soll programmiert werden



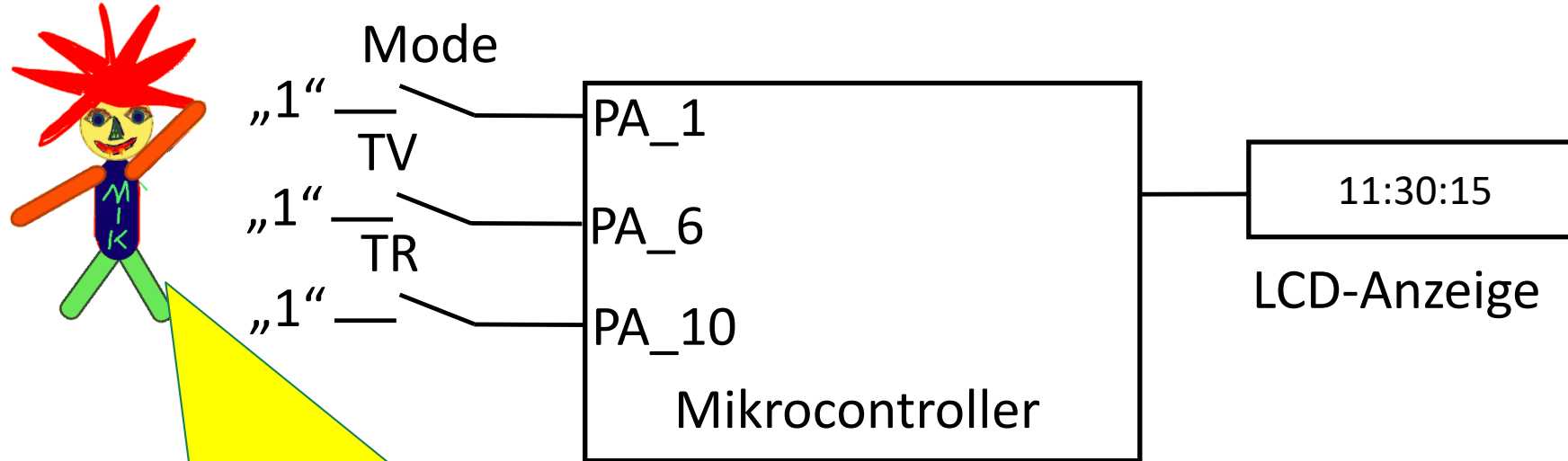
Ereignisse im Zustandsdiagramm



Die Taster benötigen PullDown und sind als Interrupteingänge zu konfigurieren



Ereignisse im Zustandsdiagramm

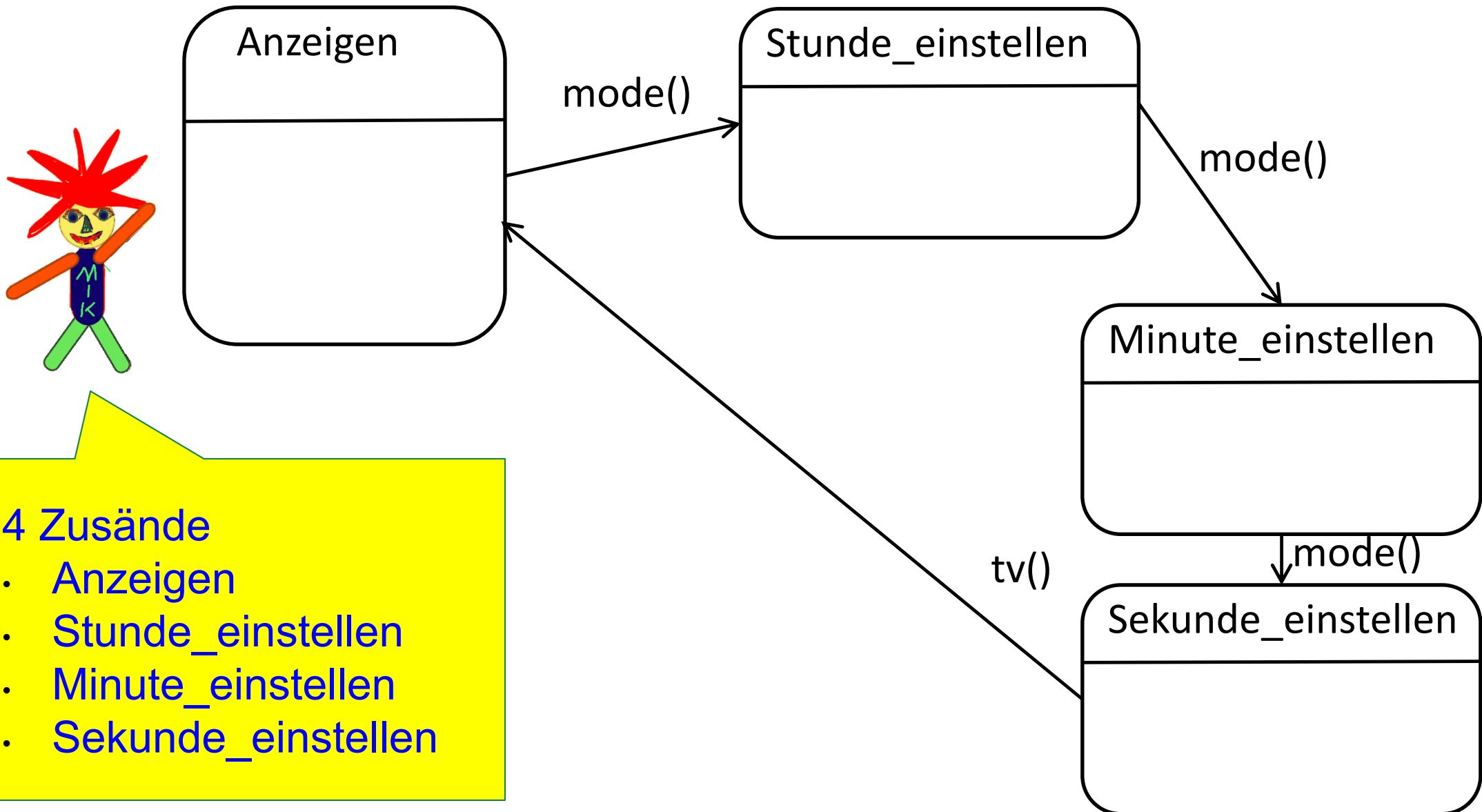


Taste „mode“ wechselt die Betriebsarten:

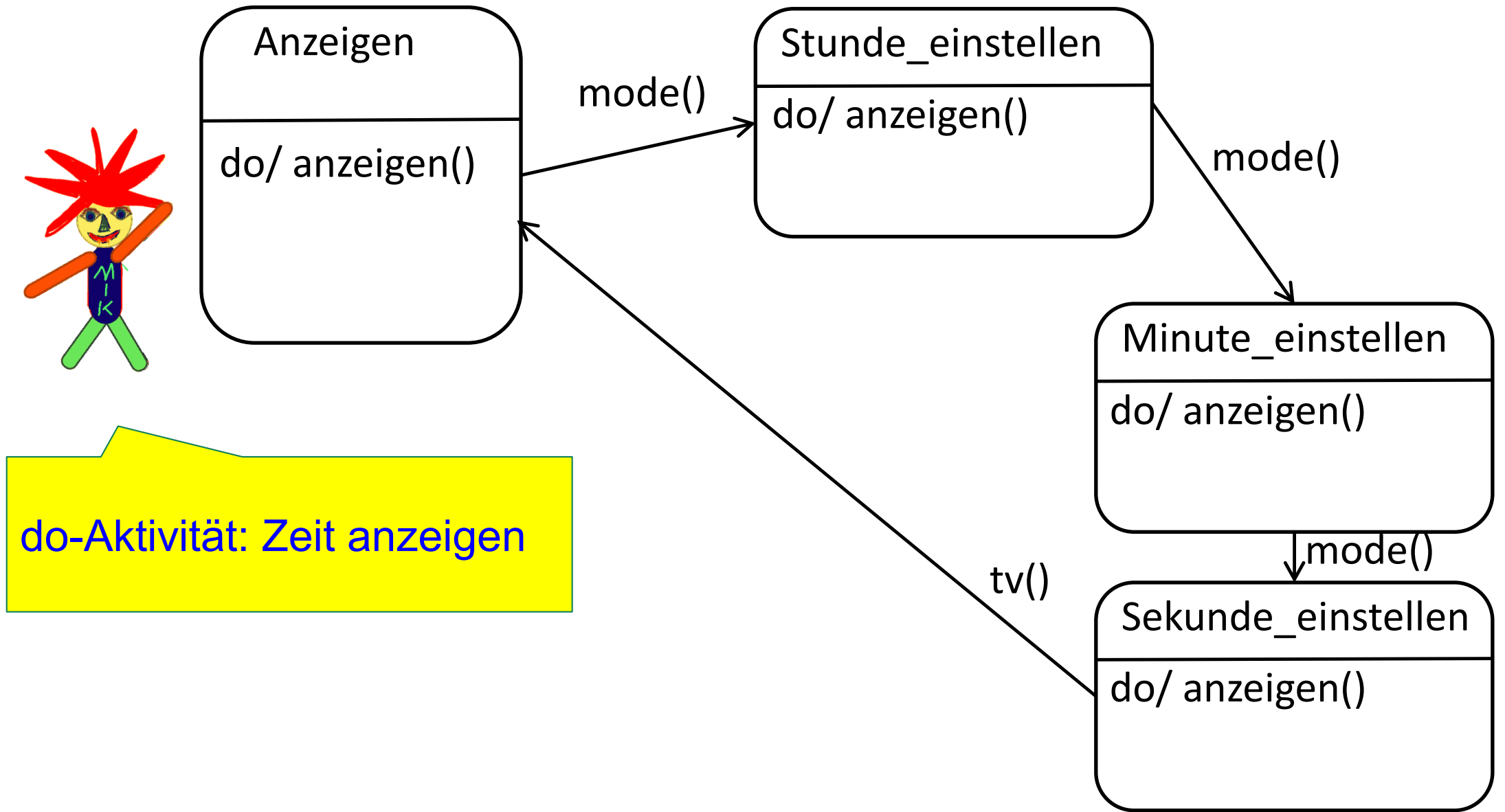
- Anzeigen
- Stunde_einstellen
- Minute_einstellen
- Sekunde_einstellen



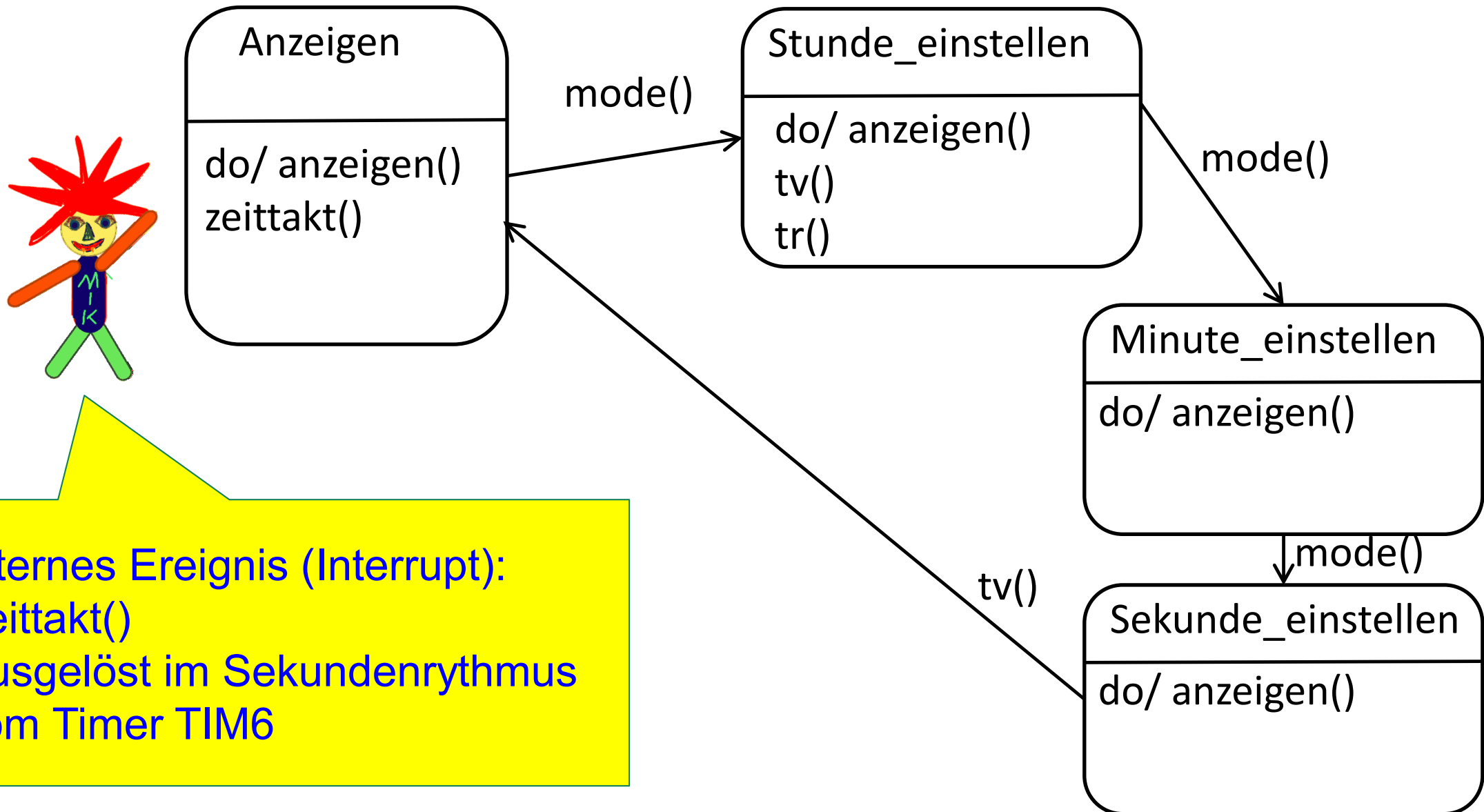
Ereignisse im Zustandsdiagramm



Ereignisse im Zustandsdiagramm



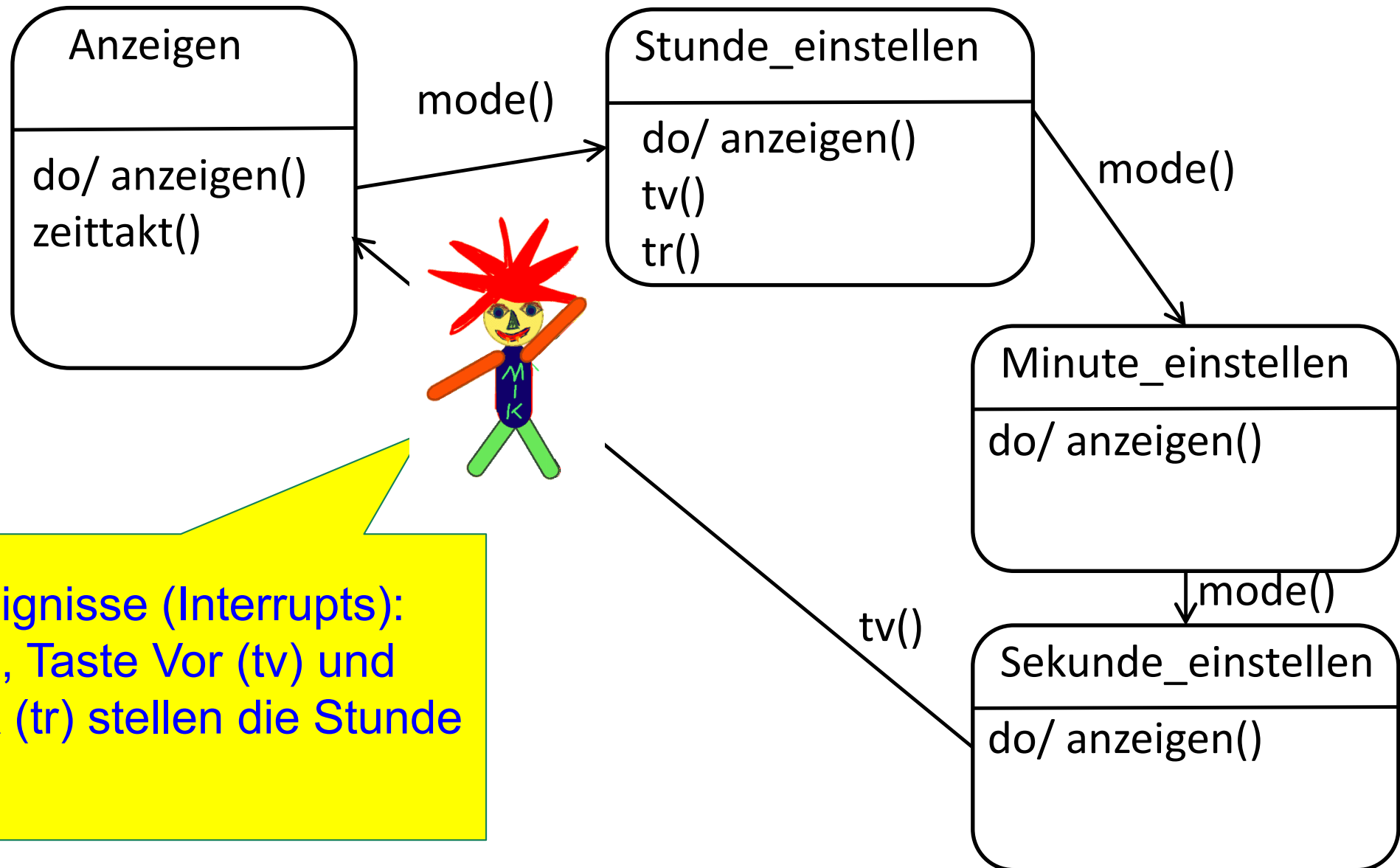
Ereignisse im Zustandsdiagramm



Internes Ereignis (Interrupt):
Zeittakt()
Ausgelöst im Sekundenrhythmus
vom Timer TIM6



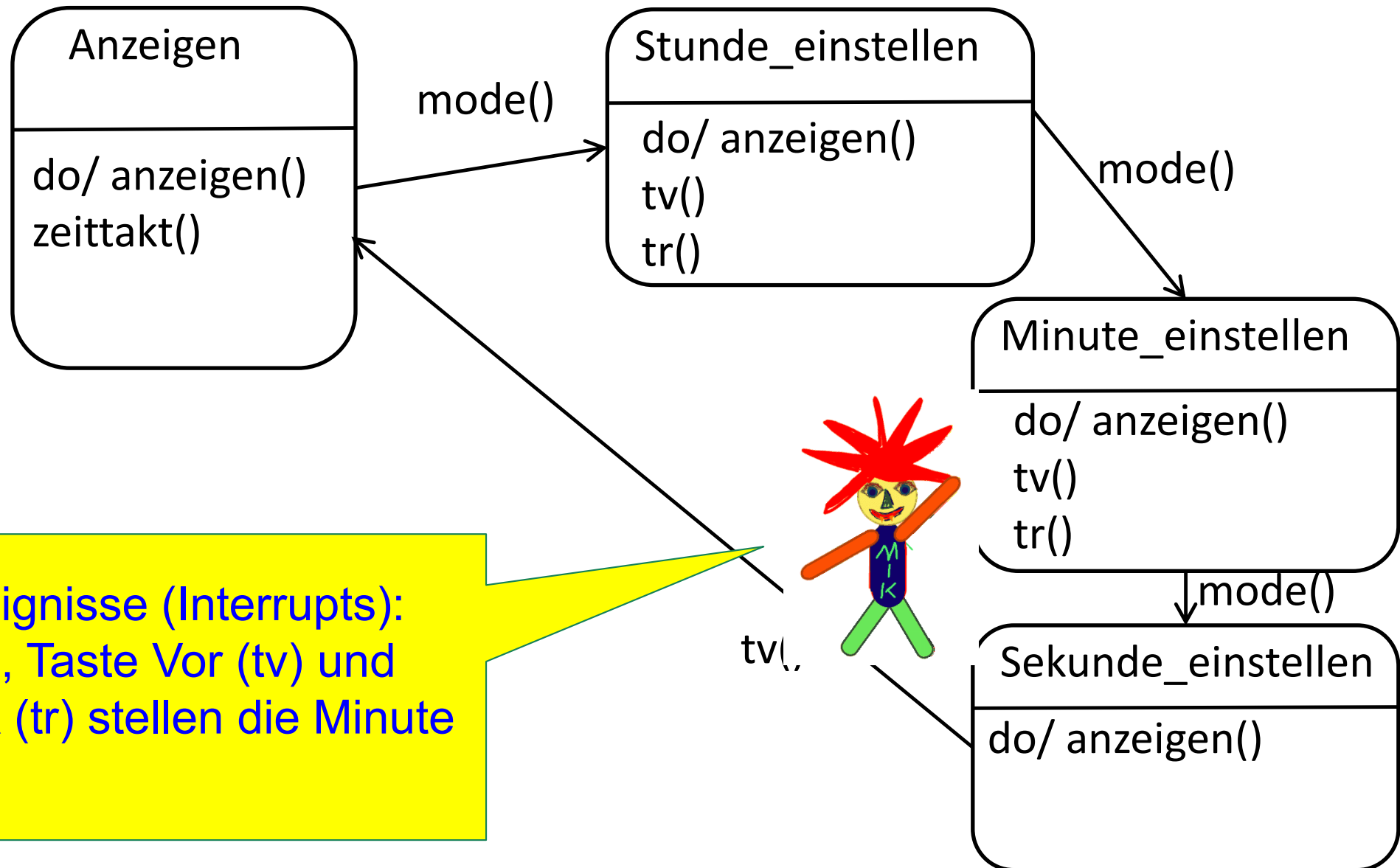
Ereignisse im Zustandsdiagramm



Interne Ereignisse (Interrupts):
tv() und tr(), Taste Vor (tv) und
Taste Rück (tr) stellen die Stunde
ein.



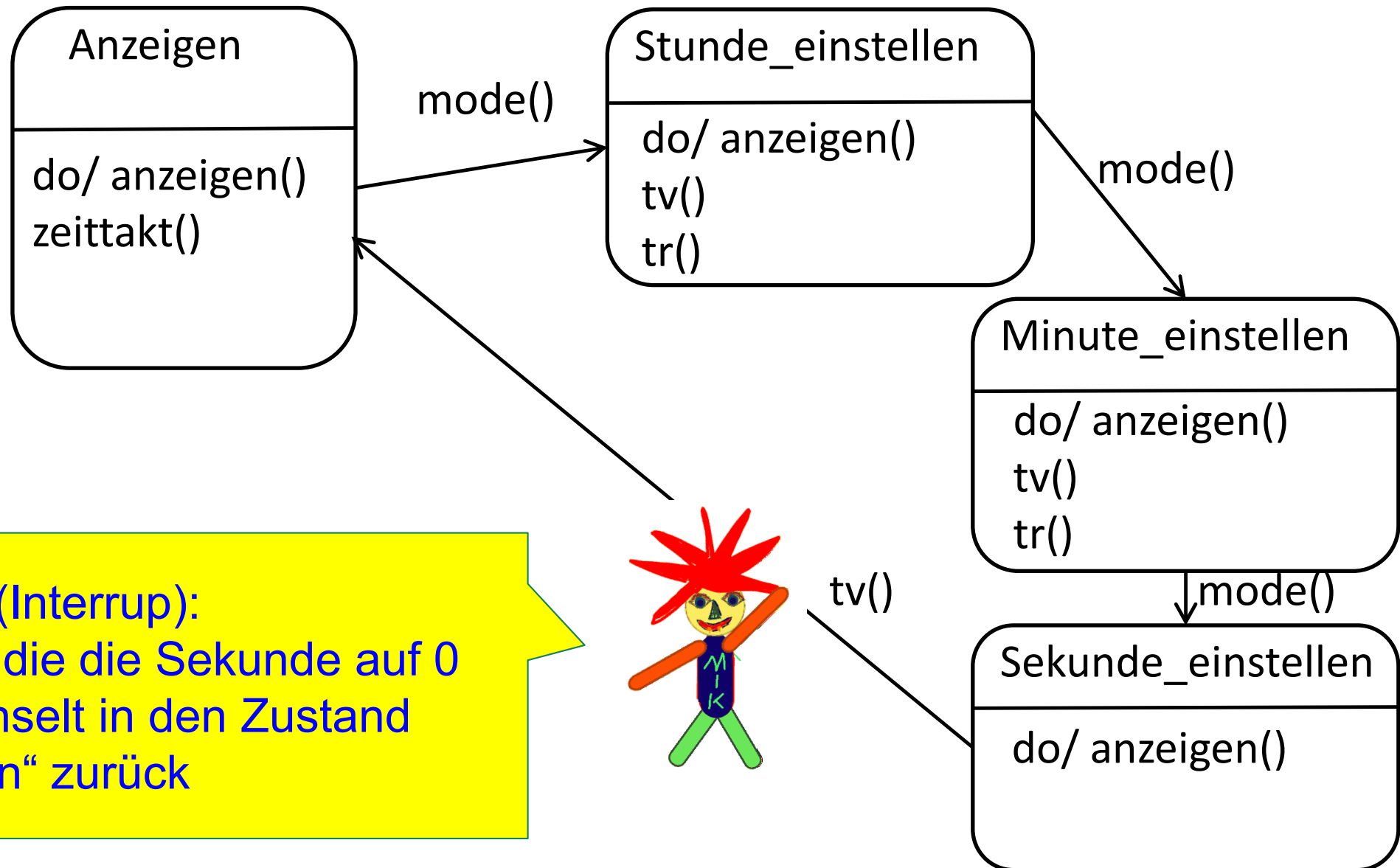
Ereignisse im Zustandsdiagramm



Interne Ereignisse (Interrupts):
tv() und tr(), Taste Vor (tv) und
Taste Rück (tr) stellen die Minute
ein.



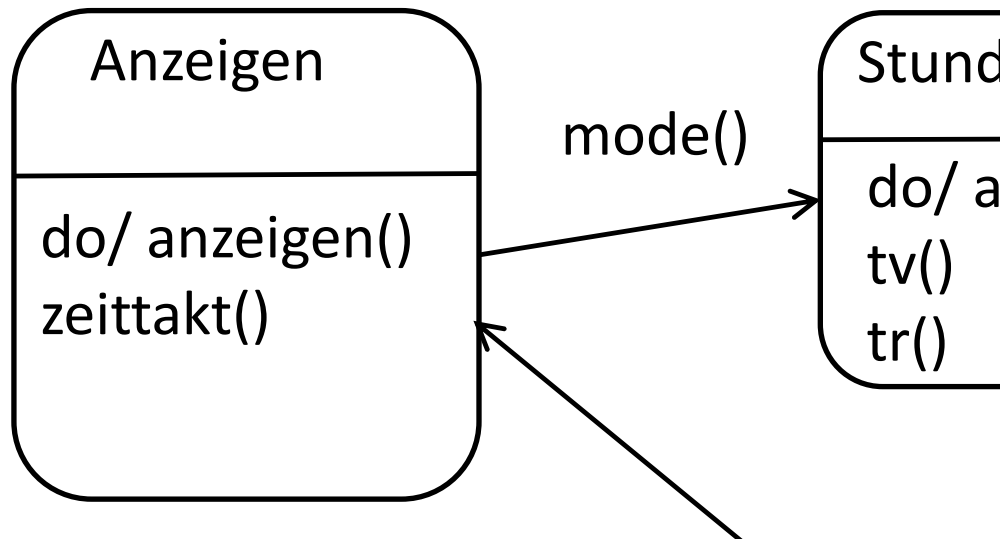
Ereignisse im Zustandsdiagramm



Ereignis (Interrupt):
tv() setzt die die Sekunde auf 0
und wechselt in den Zustand
„Anzeigen“ zurück



Ereignisse im Zustandsdiagramm



Die Zustände und die
Zustandsvariable int zustand



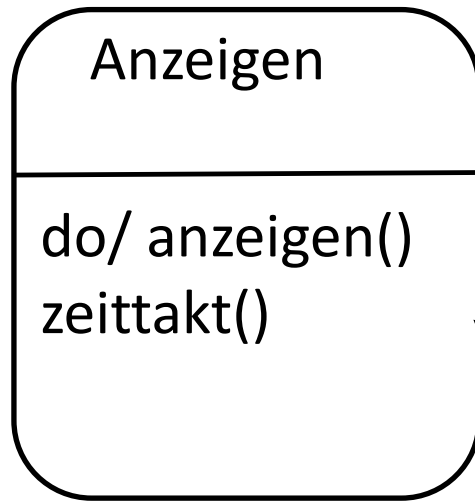
```
/* mbed Microcontroller Library
 * Copyright (c) 2019 ARM Limited
 * SPDX-License-Identifier: Apache-2.0
 */

#include "mbed.h"
#include "LCD.h",
Lcd mylcd;
#define Anzeigen 0
#define Stunde_stellen 1
#define Minute_stellen 2
#define Sekunde_stellen 3

int zustand=0;
```



Ereignisse im Zustandsdiagramm



Library LCD_i2c_GSOE in das
Projekt importieren



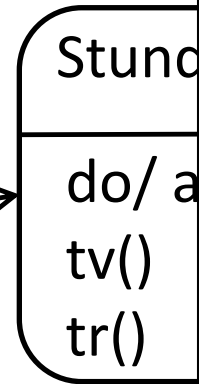
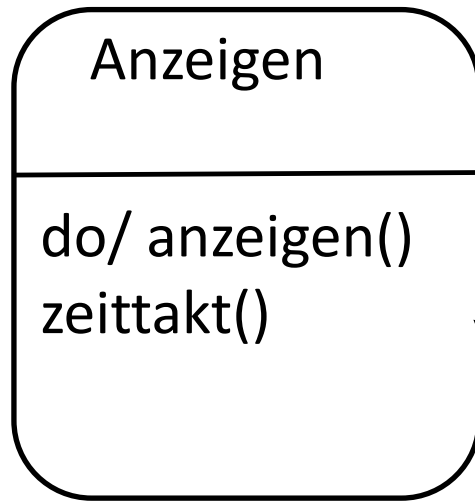
```
/* mbed Microcontroller Library
 * Copyright (c) 2019 ARM Limited
 * SPDX-License-Identifier: Apache-2.0
 */
```

```
#include "mbed.h"
#include "LCD.h",
Lcd mylcd;
#define Anzeigen 0
#define Stunde_stellen 1
#define Minute_stellen 2
#define Sekunde_stellen 3
```

```
int zustand=0;
```



Ereignisse im Zustandsdiagramm



mode()

Die Ereignisse (Interrupts):
Alle Tasten mit Pulldown
ISRs: mode(), tv(), tr()



```
InterruptIn Mode(PA_1);  
InterruptIn TV(PA_6);  
InterruptIn TR(PA_10);  
  
void initInterrupt()  
{  
    Mode.mode(PullDown);  
    Mode.rise(&mode);  
    TV.mode(PullDown);  
    TV.rise(&tv);  
    TR.mode(PullDown);  
    TR.rise(&tr);  
}
```



Ereignisse im Zustandsdiagramm

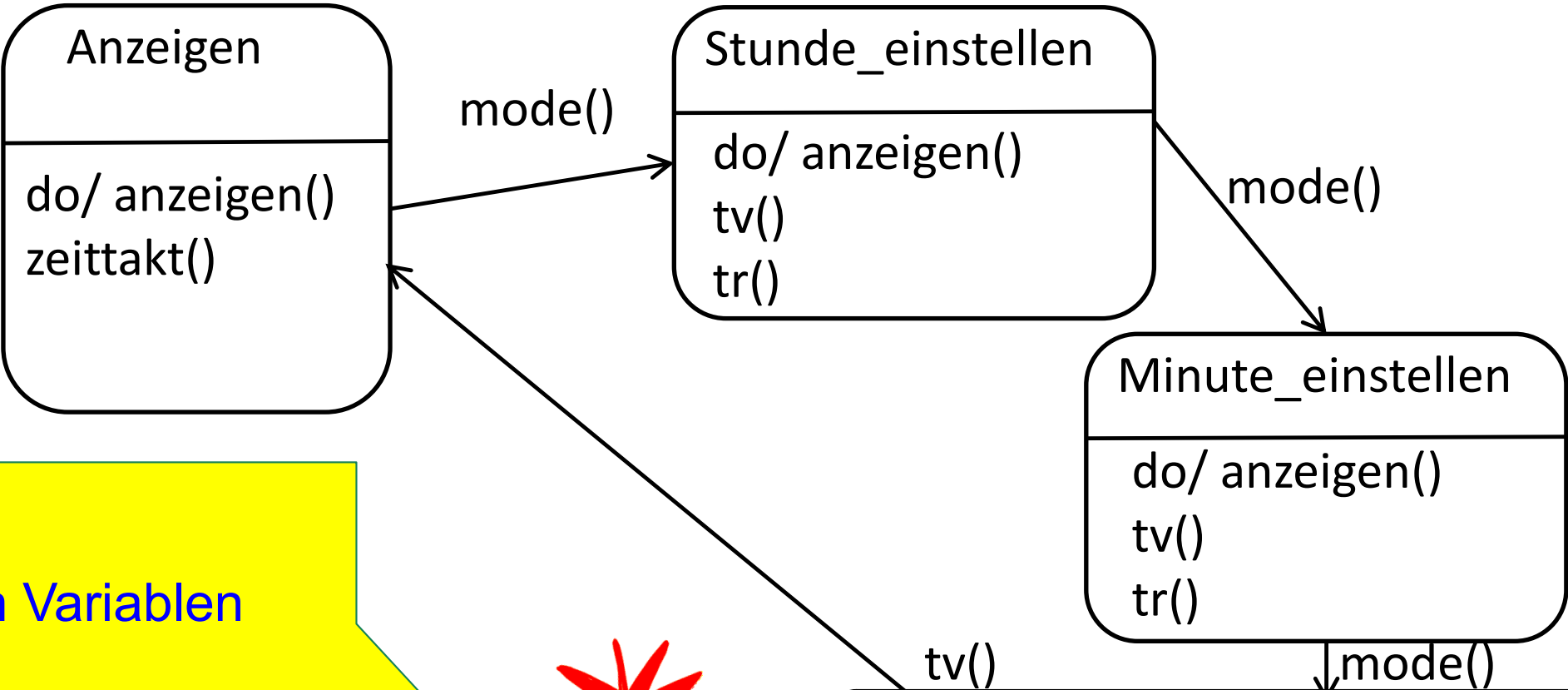


Zeitereigniss (Interrupt):
Alle 1 Sekunde
ISR: zeittakt()

```
void initTimer()
{
    RCC->APB1ENR|=0b10000; //Clock Enable
    TIM6->PSC=31999;        //Prescaler 1ms
    TIM6->ARR=1000;          //Autoreload 1000*1ms = 1s
    TIM6->DIER=1;            //UIE = 1 (Update Interrupt Enable)
    TIM6->SR=0;              //UIF =0 (Update Interrupt Flag)
    TIM6->CR1=1;             //CEN=1 (Counter Enable)
    /* TIM6_IRQn interrupt configuration */
    NVIC_SetVector(TIM6_IRQn, (uint32_t)&zeittakt);
    HAL_NVIC_EnableIRQ(TIM6_IRQn);
}
```



Ereignisse im Zustandsdiagramm



Die globalen Variablen

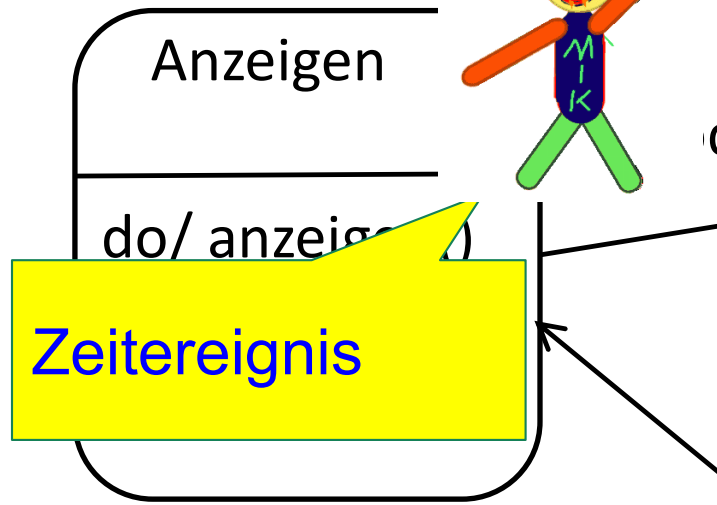


1

int stunde=23, minute=59, sekunde=0;

2

Ereignisse im Zustandsdiagramm:



```
void zeittakt()
```

```
{ if (zustand==Anzeigen)
```

```
{ sekunde++;
```

```
  if (sekunde==60)
```

```
  { sekunde=0;
```

```
    minute++;
```

```
    if (minute==60)
```

```
    { minute=0;
```

```
      stunde++;
```

```
      if (stunde==24)
```

```
      { stunde=0;
```

```
      }
```

```
    } }
```

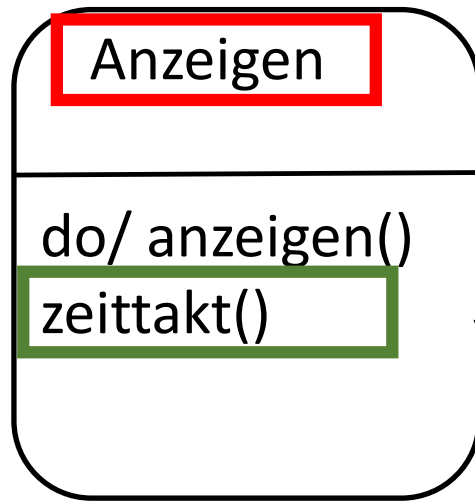
```
    TIM6->SR=0;
```

```
    HAL_NVIC_ClearPendingIRQ(TIM6_IRQn);
```

```
}
```



Ereignisse im Zustandsdiagramm



Im Zustand „Anzeigen“ wird die Zeit im Sekundentakt hochgezählt

```
void zeittakt()
```

```
{ if (zustand==Anzeigen)
```

```
{ sekunde++;
```

```
  if (sekunde==60)
```

```
  { sekunde=0;
```

```
    minute++;
```

```
    if (minute==60)
```

```
    { minute=0;
```

```
      stunde++;
```

```
      if (stunde==24)
```

```
      { stunde=0;
```

```
      }
```

```
    } }
```

```
    TIM6->SR=0;
```

```
    HAL_NVIC_ClearPendingIRQ(TIM6_IRQn);
```

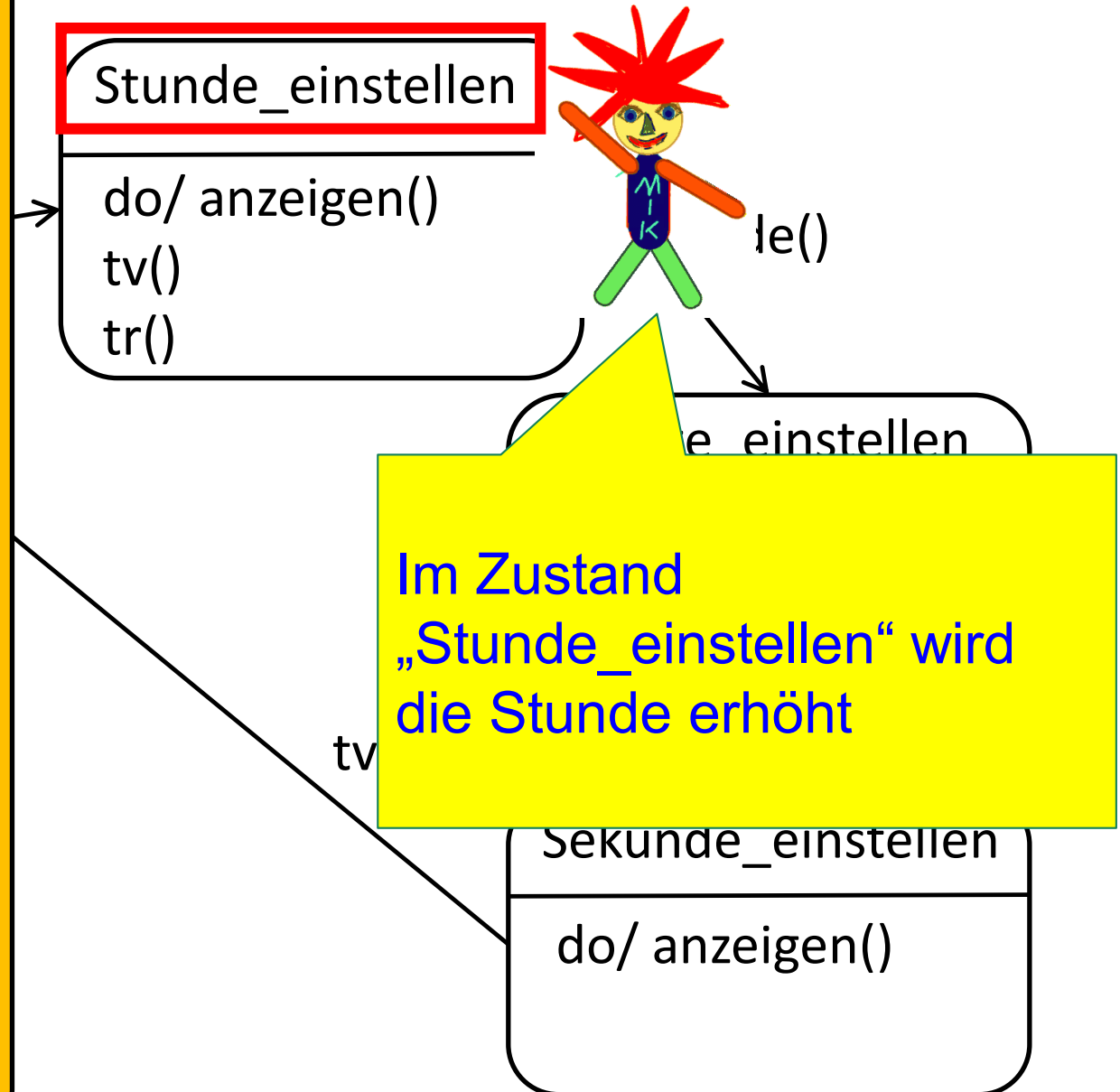
```
}
```



```

void tv()
{
  switch (zustand)
  {
    case stunde_stellen: stunde++;
      if (stunde==24) stunde=0;
      break;
    case minute_stellen: minute++;
      if (minute==60) {
        minute=0; stunde++;
        if (stunde==24) stunde=0;
      }
      break;
    case sekunde_stellen: sekunde=0;
      zustand=Anzeigen;
      break;
  }
}

```



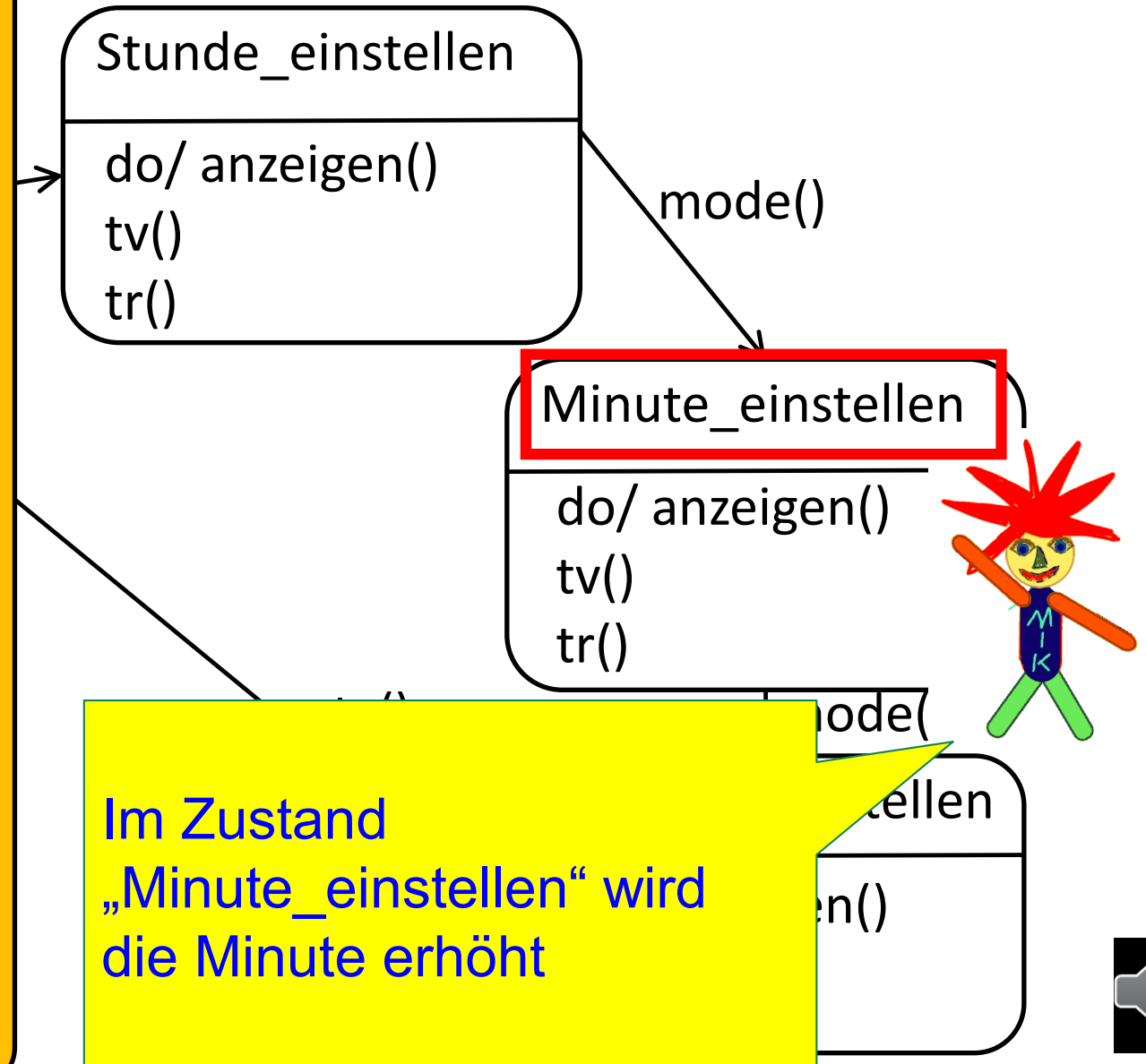
```

void tv()
{
    switch (zustand)
    {
        case stunde_stellen: stunde++;
            if (stunde==24) stunde=0;
            break;

        case minute_stellen: minute++;
            if (minute==60) {
                minute=0; stunde++;
                if (stunde==24) stunde=0;
            }
            break;

        case sekunde_stellen: sekunde=0;
            zustand=Anzeigen;
            break;
    }
}

```



```

void tv()
{
    switch (zustand)
    {
        case stunde_stellen: stunde++;
            if (stunde==24) stunde=0;
            break;
        case minute_stellen: minute++;
            if (minute==60) {
                minute=0; stunde++;
                if (stunde==24) stunde=0;
            }
            break;
        case sekunde_stellen: sekunde=0;
            zustand=Anzeigen;
            break;
    }
}

```

Im Zustand
„Sekunde_einstellen“ wird
die Sekunde auf 0 gesetzt
und in den Zustand
„Anzeigen gewechselt

Minute_einstellen

do/ anzeigen()
tv()
tr()

↓mode(

Sekunde_einstellen

do/ anzeigen()

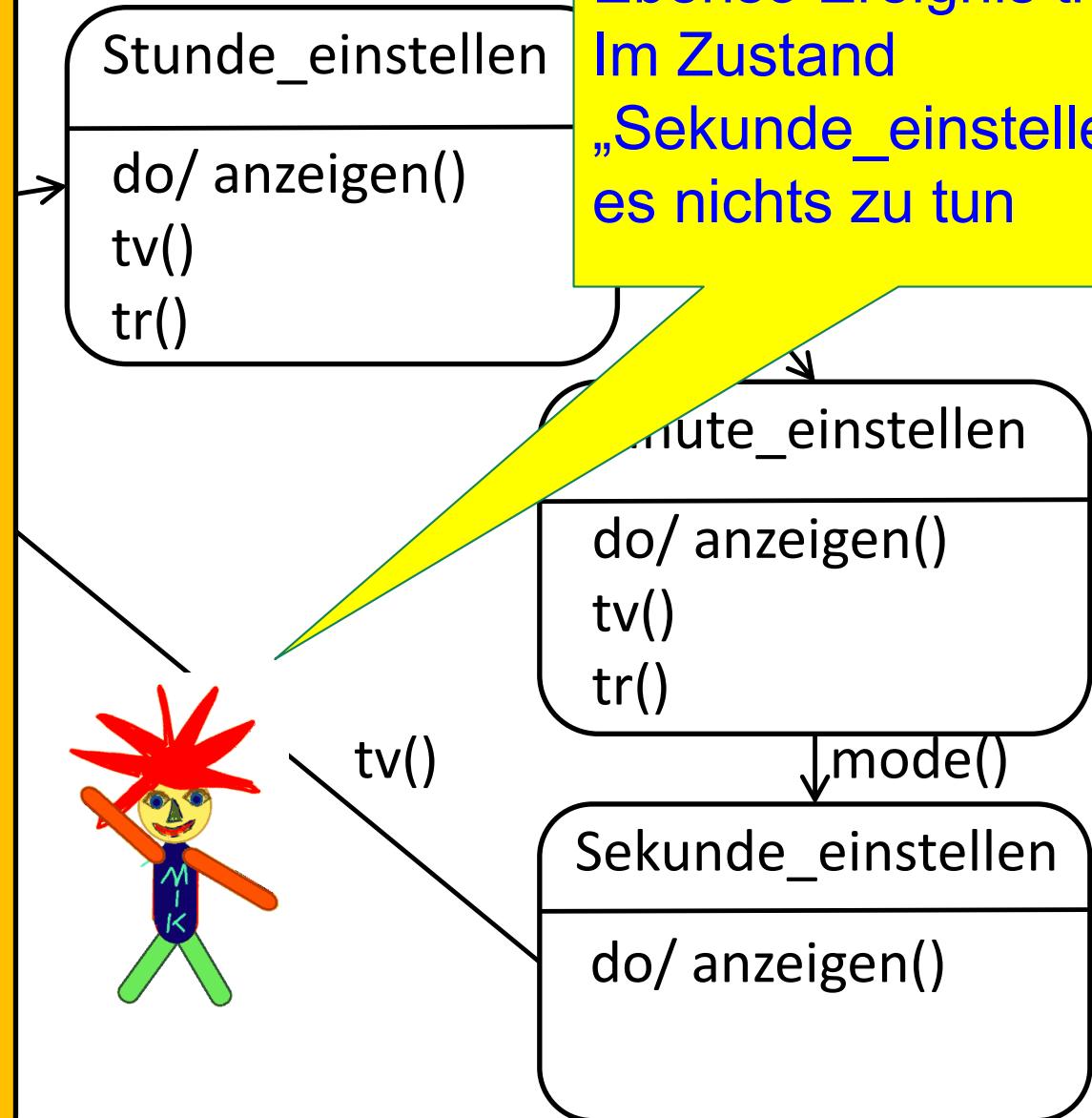
tv()



```

void tr()
{
    switch (zustand)
    {
        case stunde_stellen: stunde--;
            if (stunde== -1) stunde=23;
            break;
        case minute_stellen: minute--;
            if (minute== -1) {
                minute=59; stunde--;
                if (stunde== -1) stunde=23;
            }
            break;
    }
}

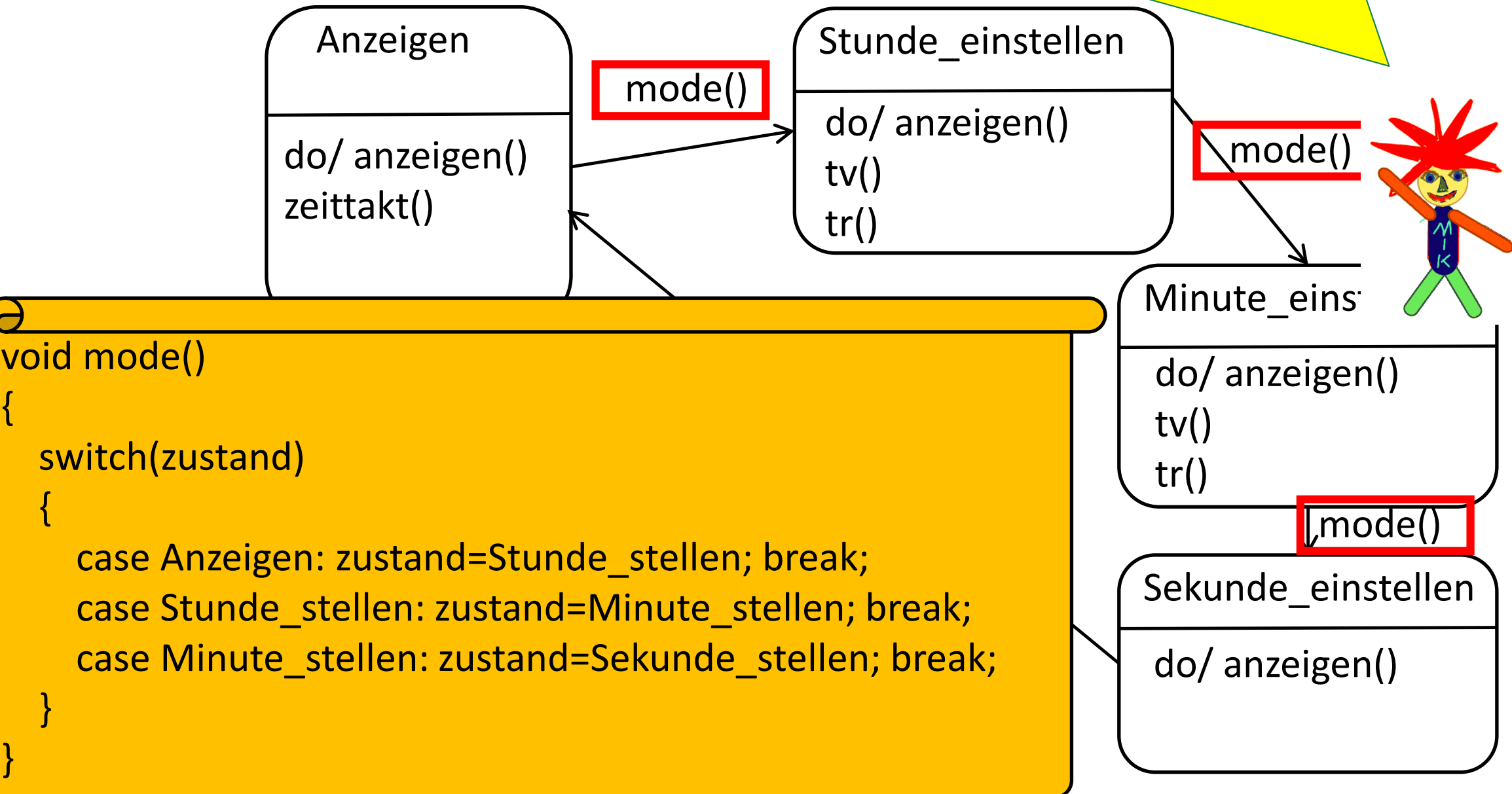
```



Ebenso Ereignis tr()
Im Zustand
„Sekunde_einstellen“ gibt
es nichts zu tun



Bei Taste Mode in den jeweiligen Folgezustand wechseln



```
void mode()
{
    switch(zustand)
    {
        case Anzeigen: zustand=Stunde_stellen; break;
        case Stunde_stellen: zustand=Minute_stellen; break;
        case Minute_stellen: zustand=Sekunde_stellen; break;
    }
}
```



```
int main()
```

```
{
```

```
    initTimer();
```

```
    initInterrupt();
```

```
    mylcd.clear();
```

```
    while (true) {
```

```
        switch(zustand)
```

```
        {
```

```
            case Anzeigen:
```

```
            case Stunde_stellen:
```

```
            case Minute_stellen:
```

```
            case Sekunde_stellen: anzeigen();
```

```
                break;
```

```
        }
```

```
    }
```

```
}
```

Stunde_einstellen

do/ anzeigen()

tv()

tr()

Hauptprogramm:

Alles Initialisieren und
In allen Zuständen die do-
Aktivität „anzeigen()“
ausführen

Minute_einstellen

do/ anzeigen()

tv()

tr()

mode()

Sekunde_einstellen

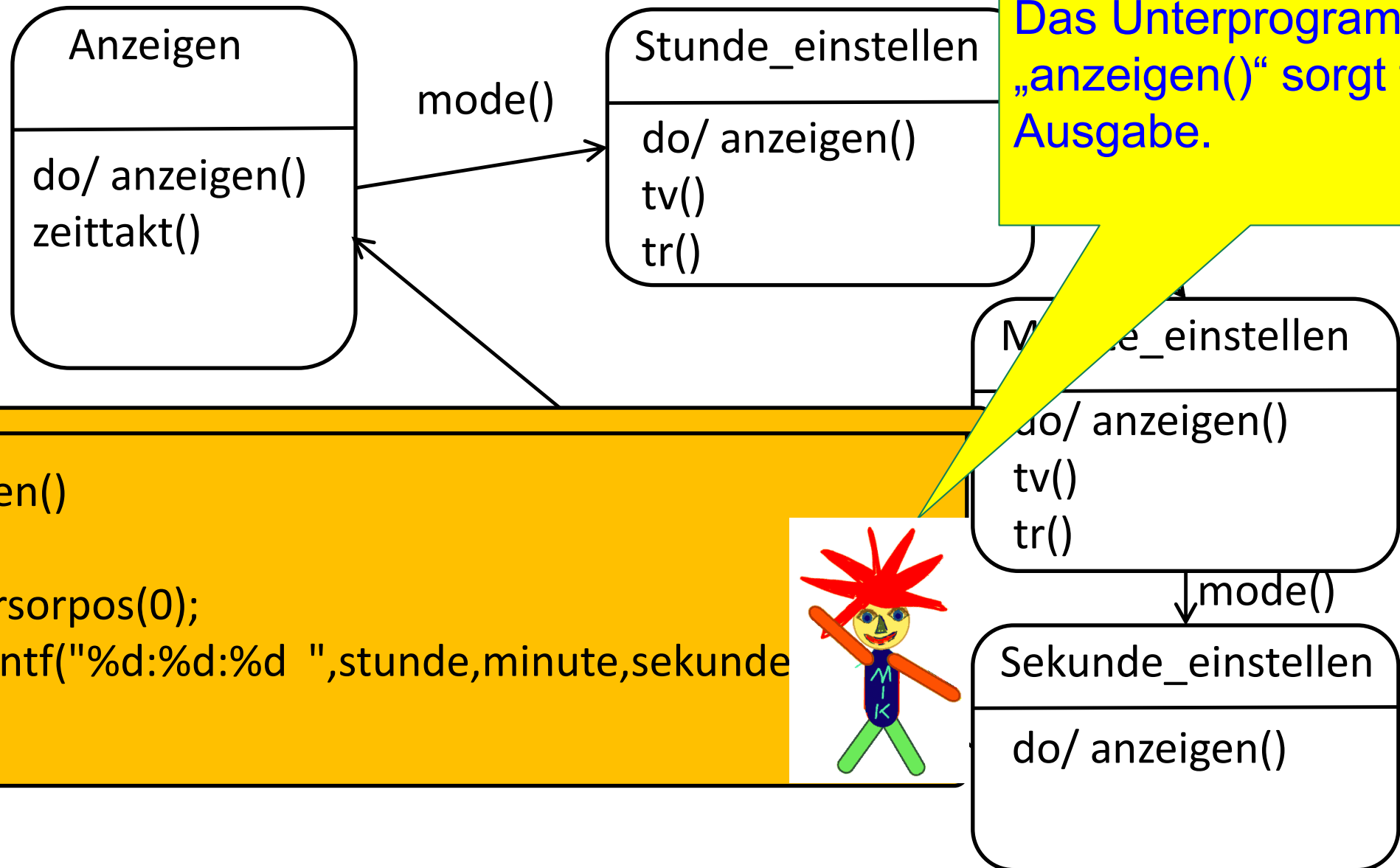
do/ anzeigen()



tv()



Ereignisse im Zustandsdiagramm



```
void anzeigen()
{
    mylcd.cursorpos(0);
    mylcd.printf("%d:%d:%d ",stunde,minute,sekunde);
}
```

