



Timer

Basic Timer TIM6 und TIM7



Ich bin Mik, Dein Mikrocontroller



Timer



Solange die Zitrone im
Körbchen sichtbar ist:
Zähle auf hundert.
Zähle genau jede
Sekunde um 1 weiter.
Wenn Du dort
angekommen bist,
beginne von vorne und
schalte das Licht ein.

Der Auftrag



Timer

Solange die Zitrone im
Körbchen sichtbar ist:
Zähle auf hundert.
Zähle genau jede
Sekunde um 1 weiter.
Wenn Du dort
angekommen bist,
beginne von vorne und
schalte das Licht ein.

Obwohl ich viel weiter
zählen kann soll ich nur bis
100 zählen. Und welche
Zitrone?

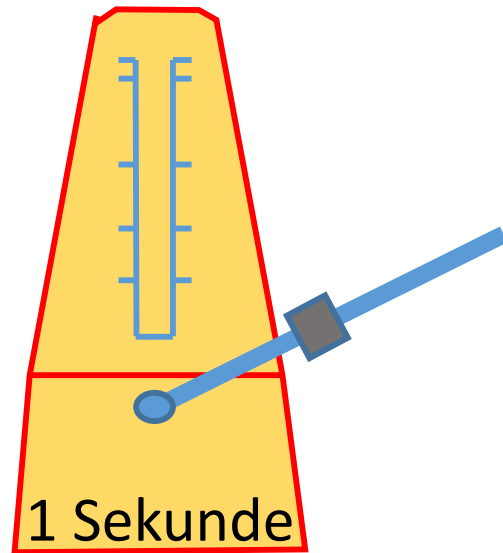


Timer



100

0



Licht

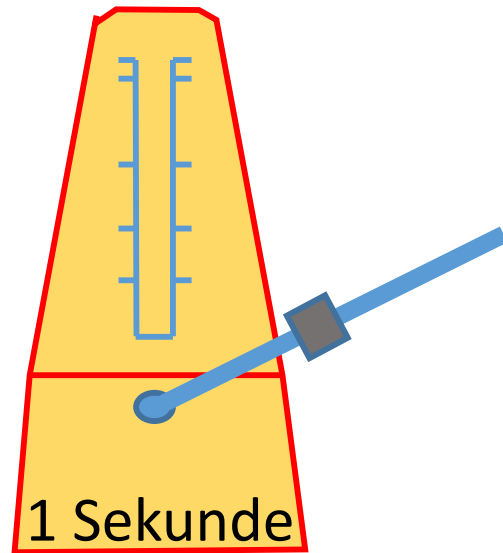


Timer



100

0



Licht

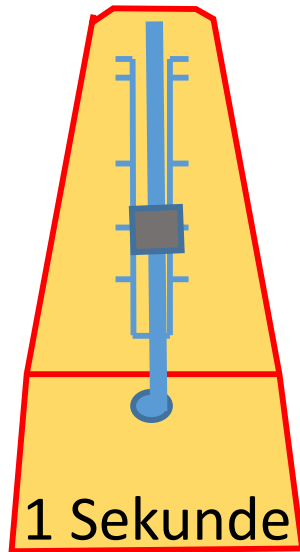


Timer

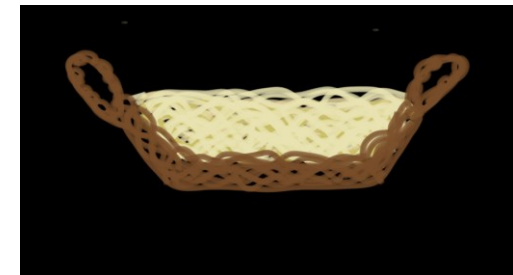


100

0



Licht

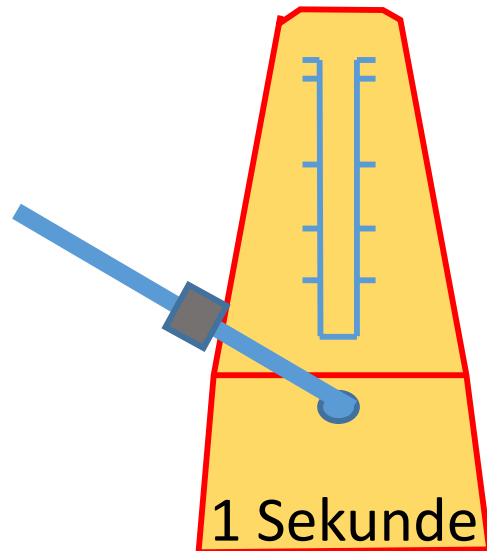


Timer



100

0



Licht

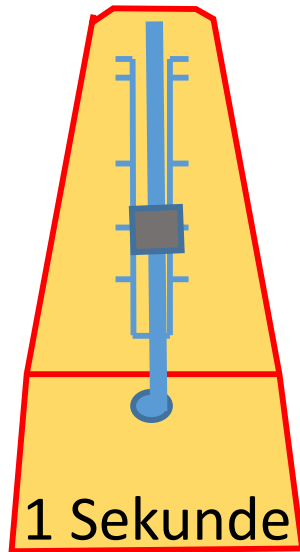


Timer



100

0



Licht

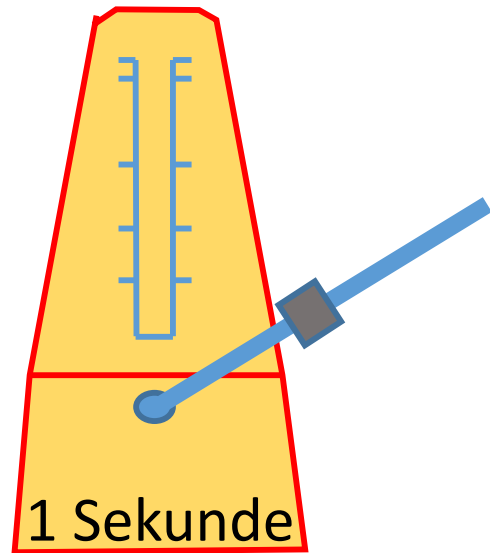


Timer



100

0



Licht

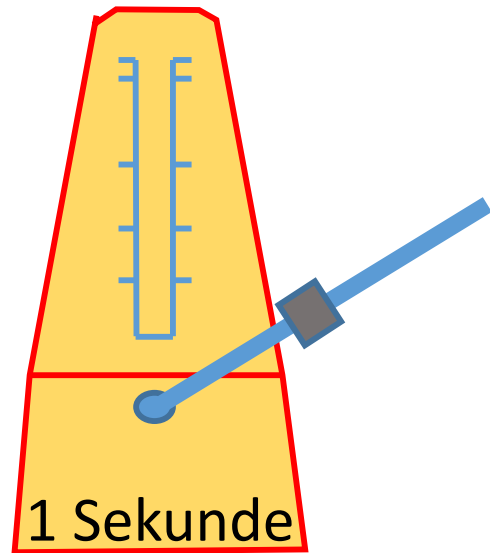


Timer



100

0



Licht

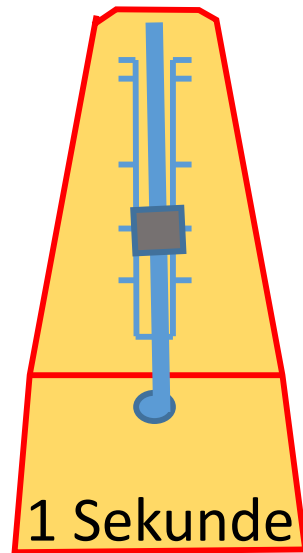


Timer



100

1



Licht

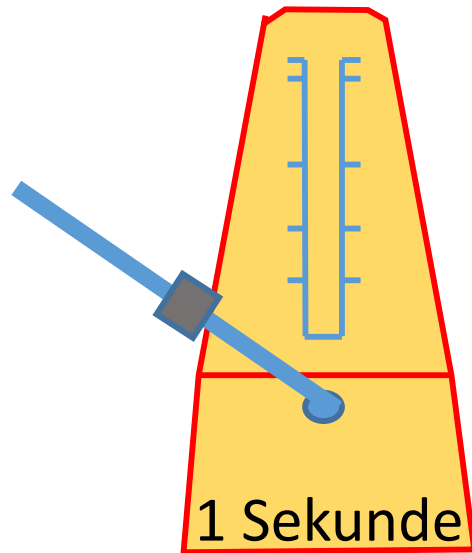


Timer



100

1



Licht

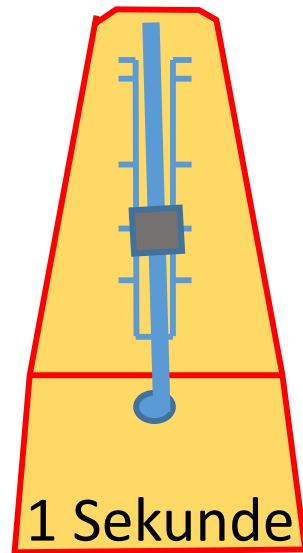


Timer



100

1



Licht

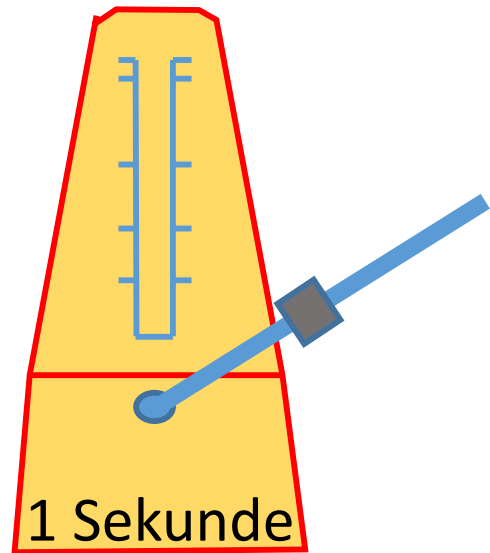


Timer



100

1



Licht

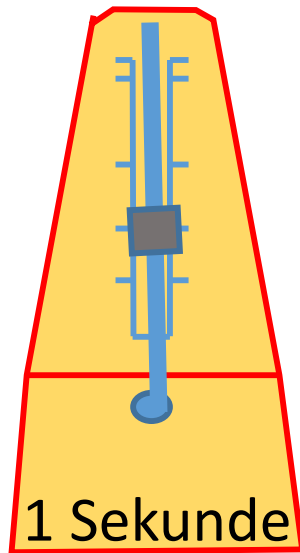


Timer



100

2



Licht

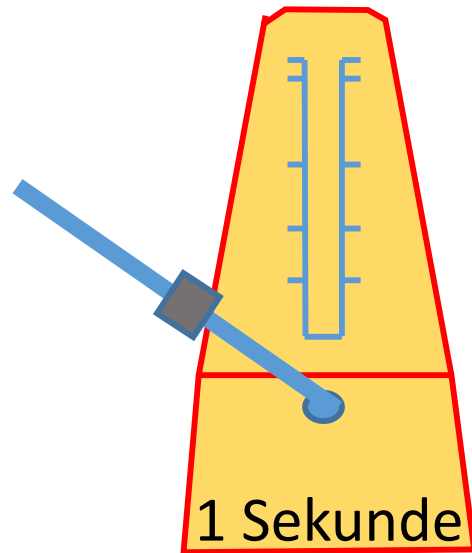


Timer



100

2



Licht

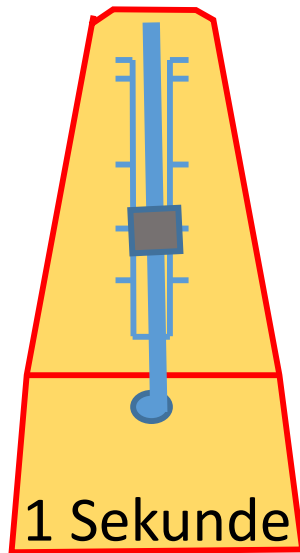


Timer



100

2



Licht

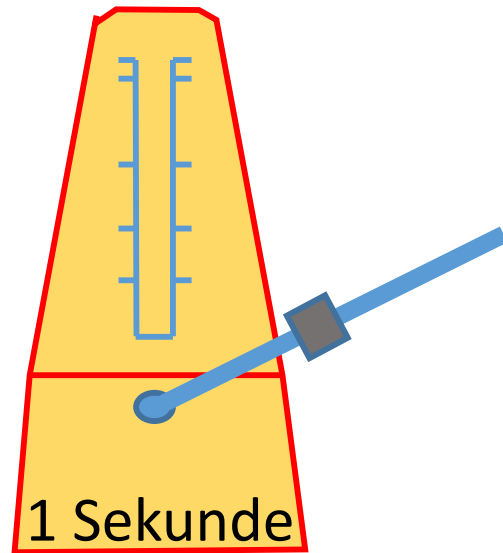


Timer



100

2



Licht

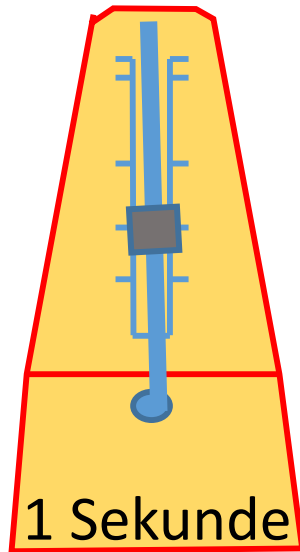


Timer



100

3



Licht

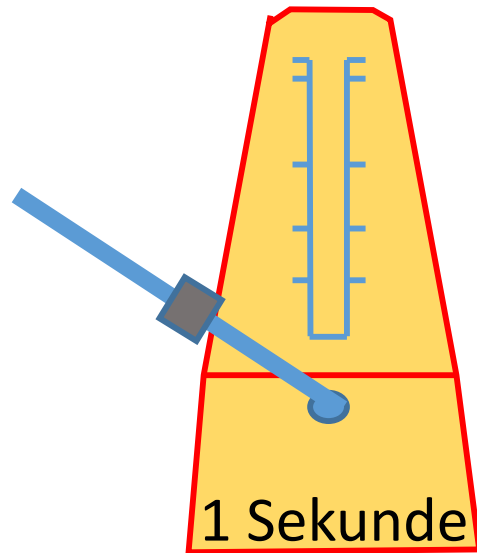


Timer



100

3



Licht

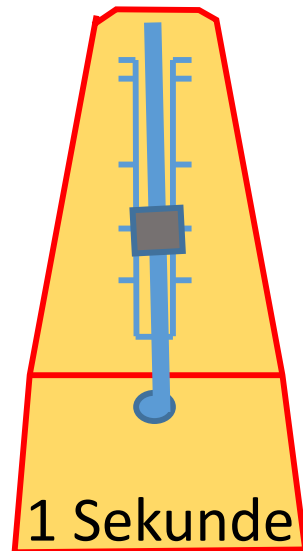


Timer



100

3



Licht

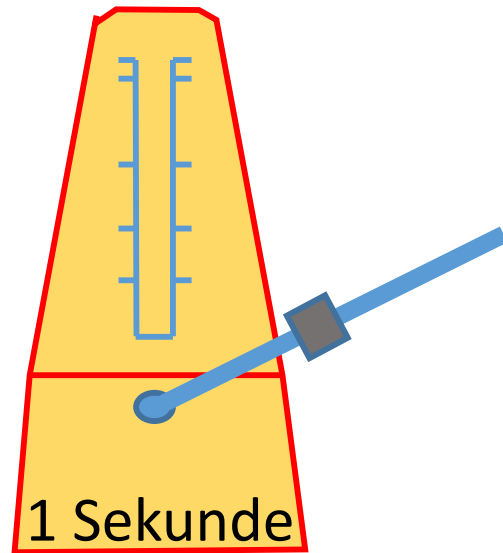


Timer



100

3



Licht

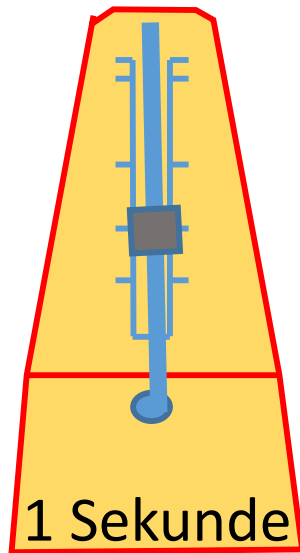


Timer



100

4



Licht



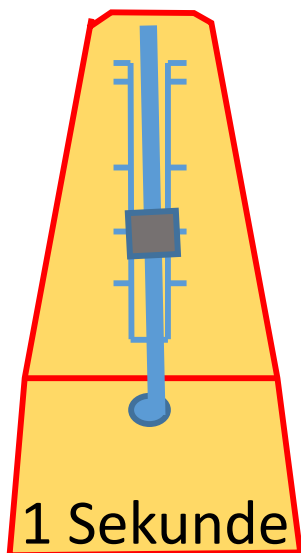
Timer



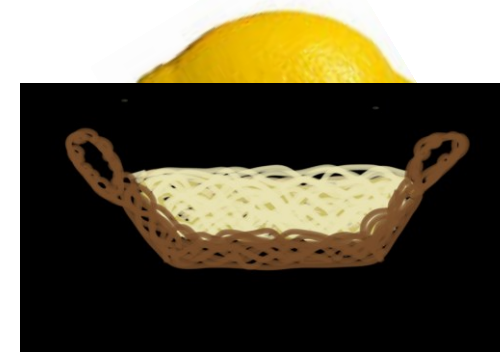
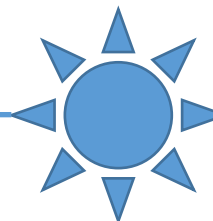
95 Sekunden später

100

99



Licht

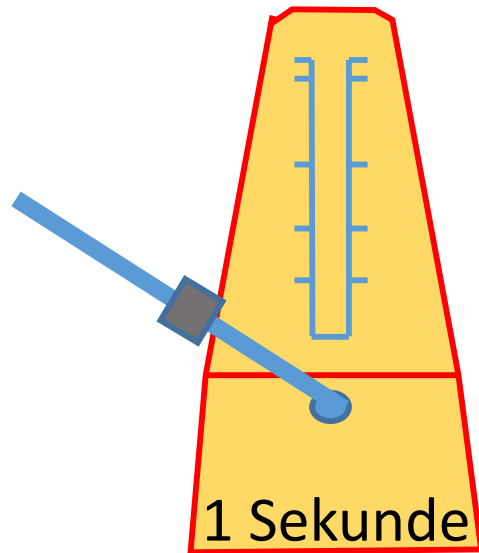


Timer



100

99



Licht

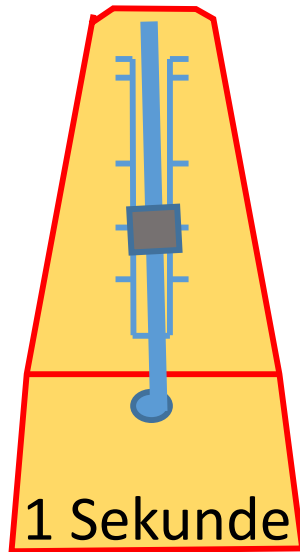


Timer



100

99



Licht

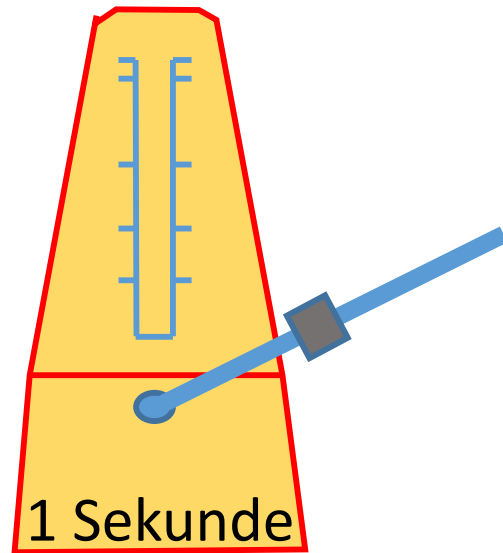


Timer



100

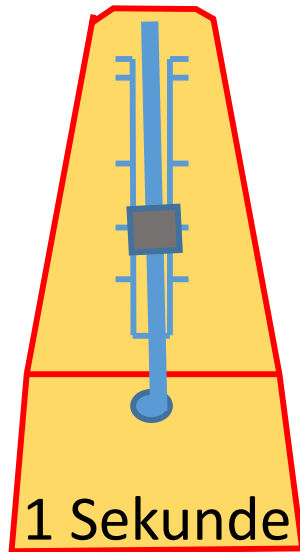
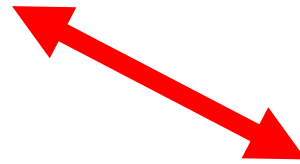
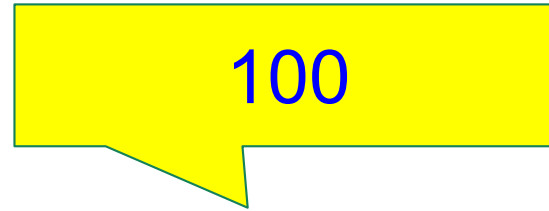
99



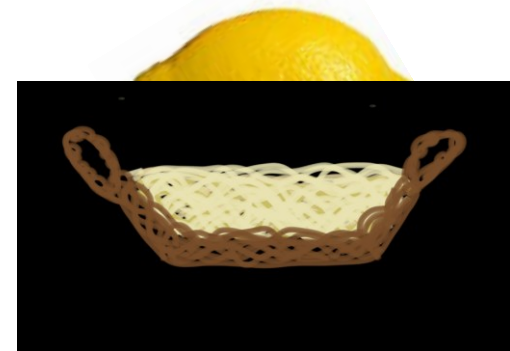
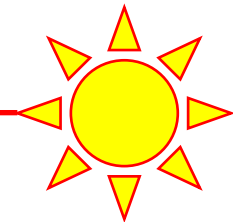
Licht



Timer



Licht

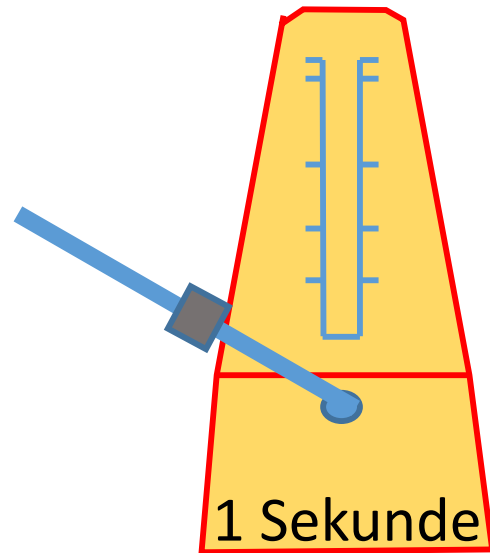


Timer



100

0



Licht

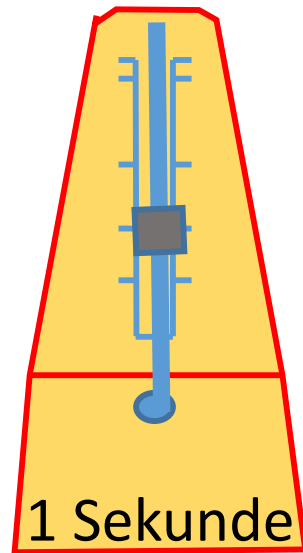


Timer



100

0



Licht

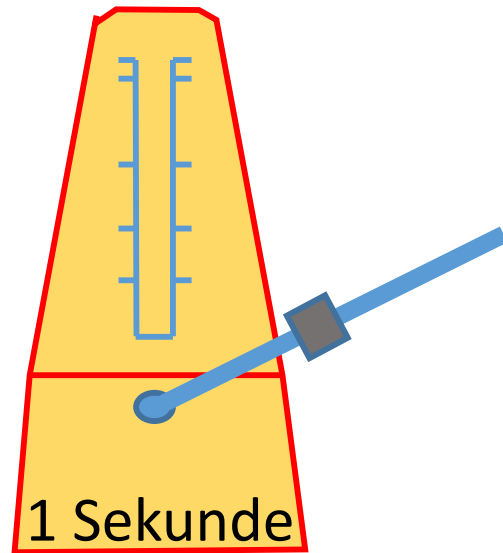


Timer



100

0



Licht

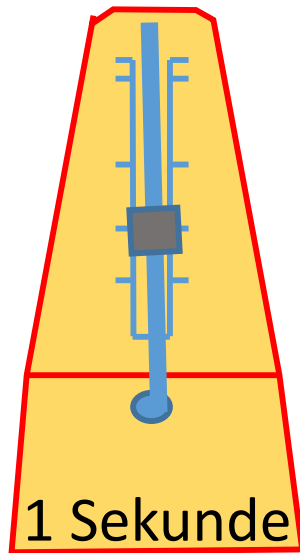


Timer



100

1



Licht

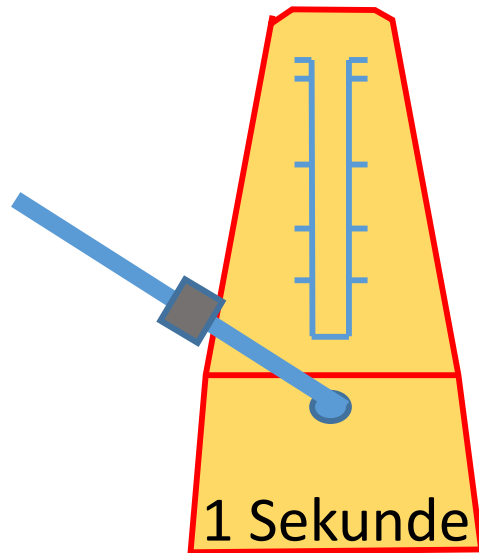


Timer



100

1



Licht

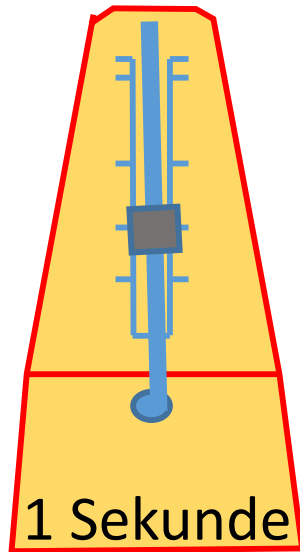


Timer



100

1



Licht

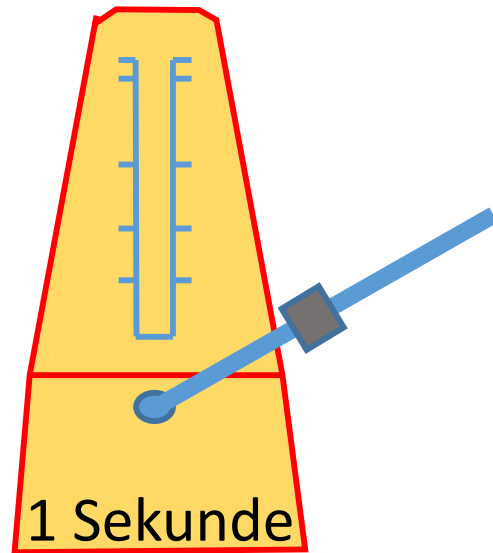


Timer



100

1



Licht

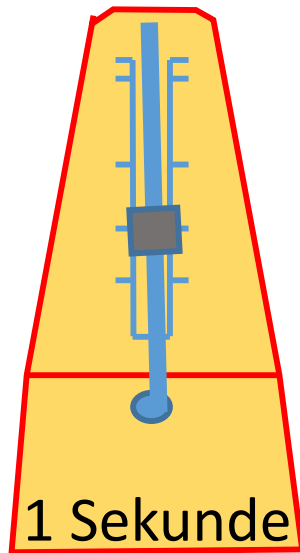


Timer



100

2



Licht

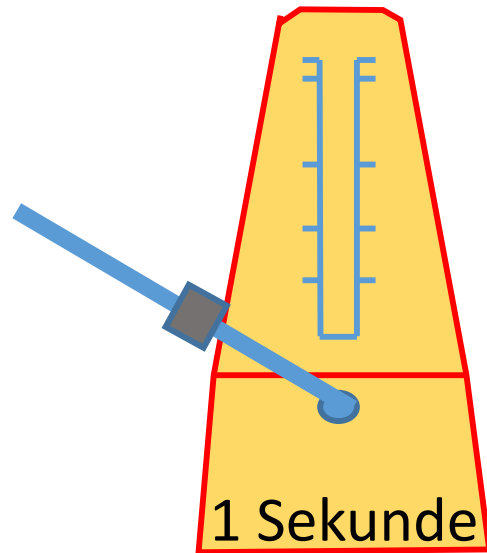


Timer



100

2



Licht

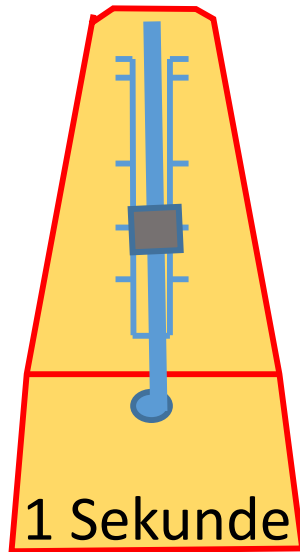


Timer



100

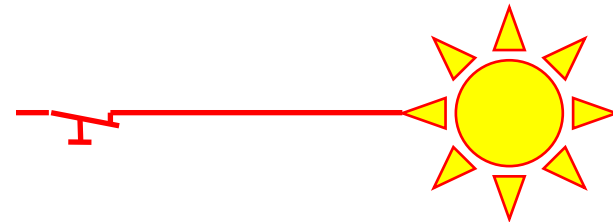
2



1 Sekunde



Licht

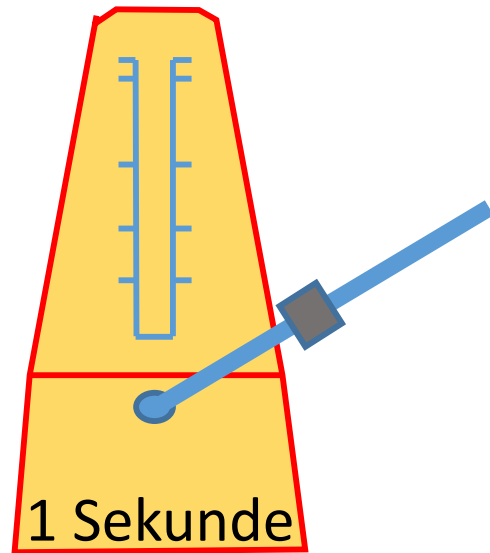


Timer



100

2



Licht

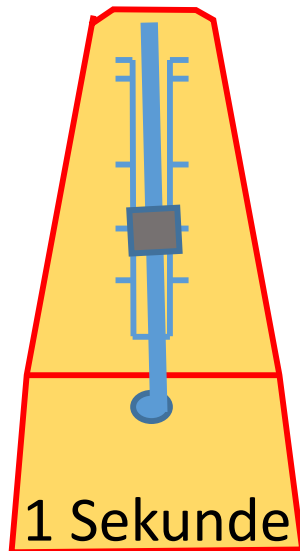


Timer



100

3



Licht

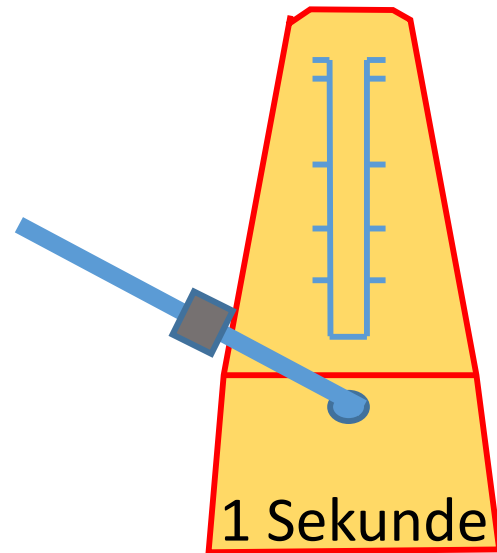


Timer

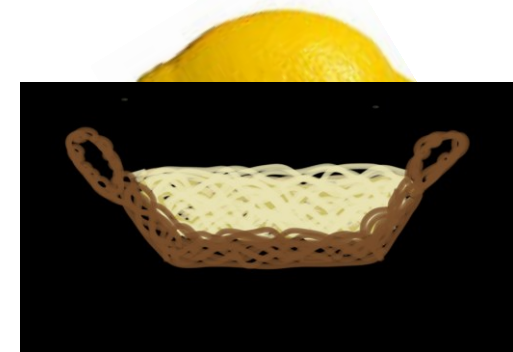


100

3



Licht

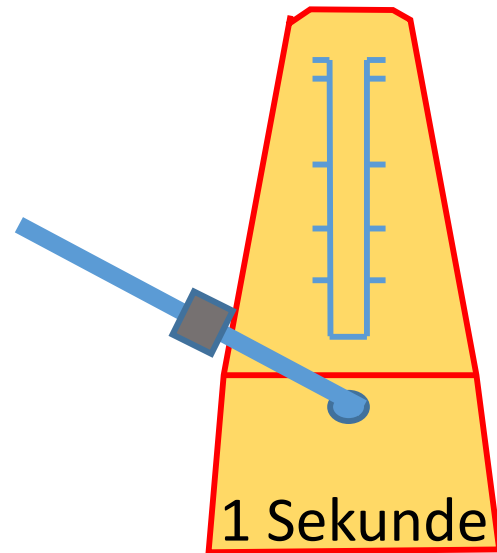


Timer



100

3



Licht

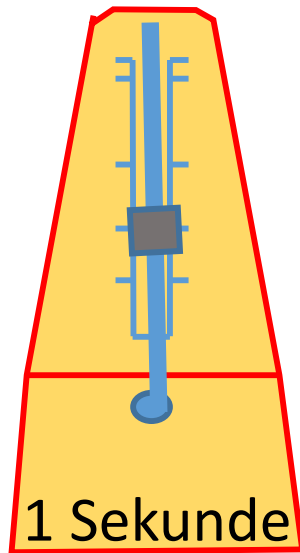


Timer



100

3



Licht

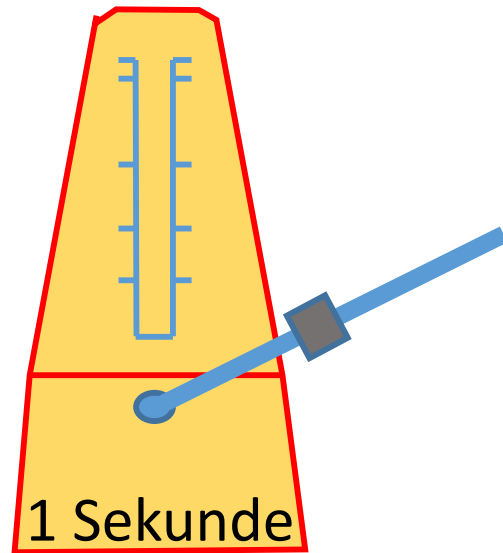


Timer

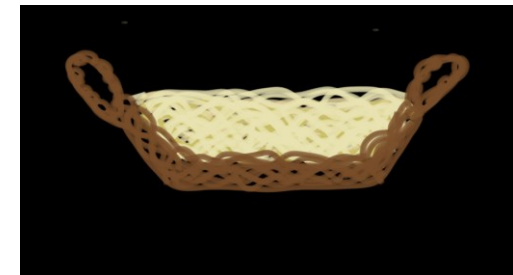


100

3



Licht

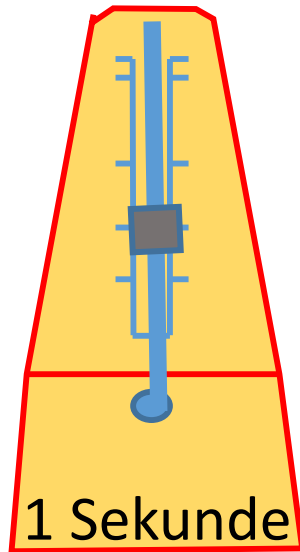


Timer



100

3



Licht

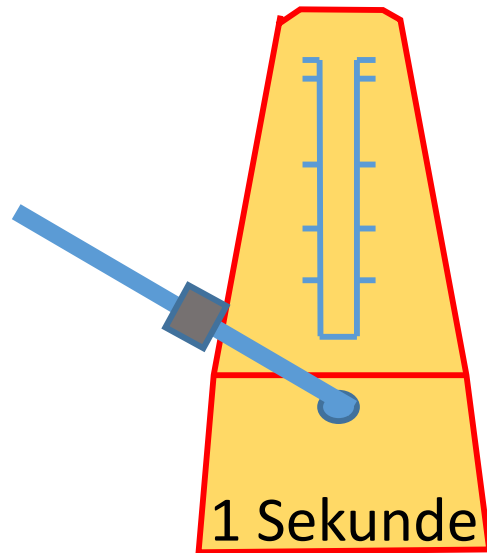


Timer



100

3



Licht

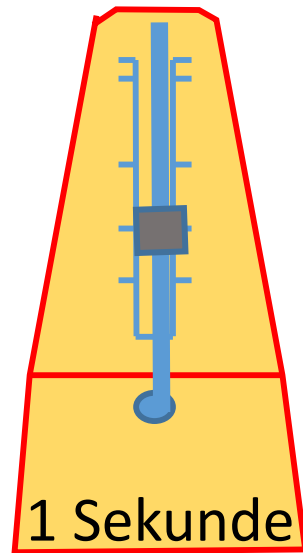


Timer



100

3



Licht

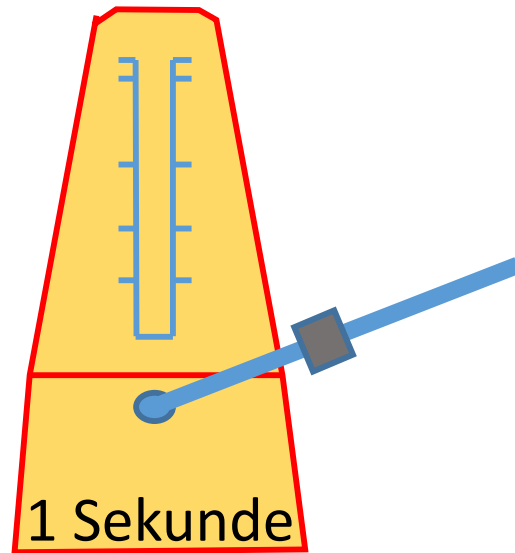


Timer

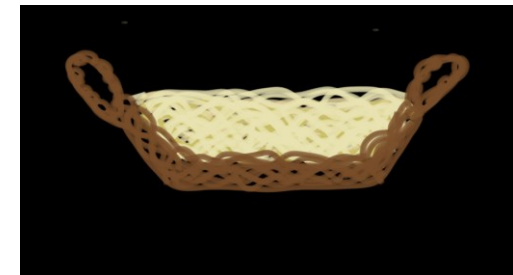


100

3



Licht

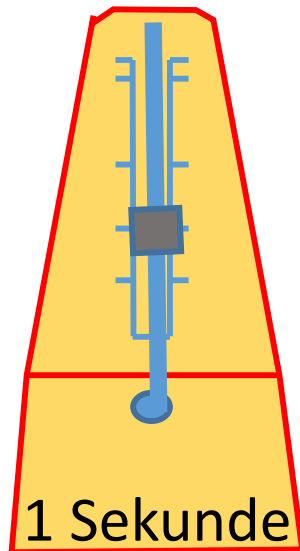


Timer



100

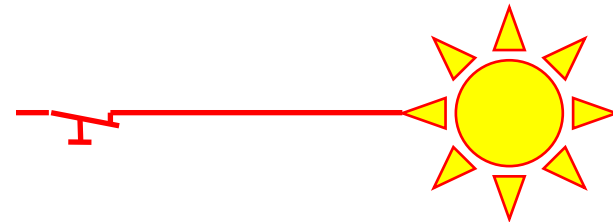
3



1 Sekunde



Licht

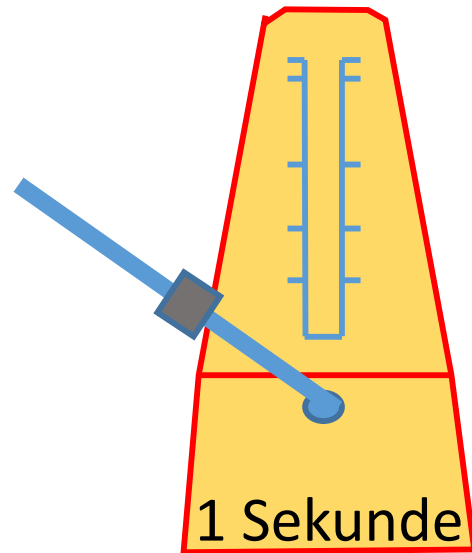


Timer

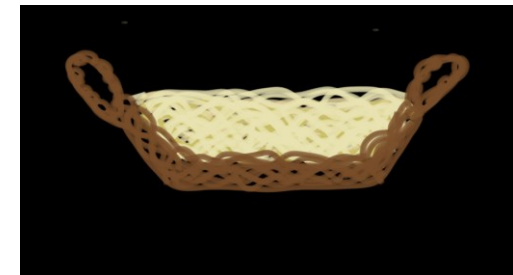


100

3



Licht

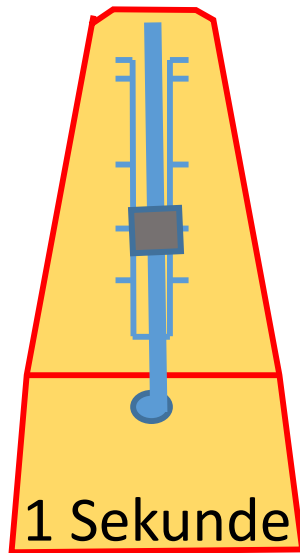


Timer



100

3



Licht

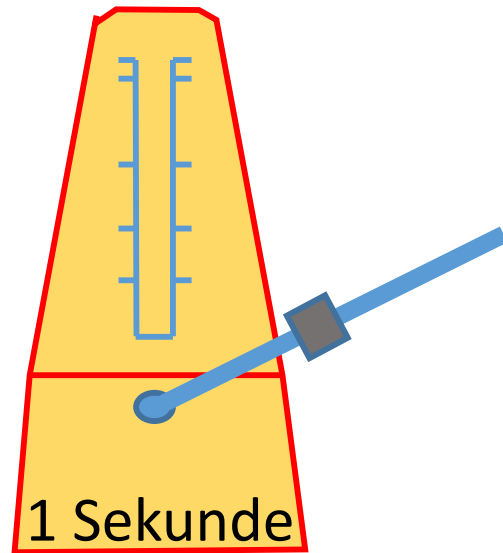


Timer



100

3



Licht

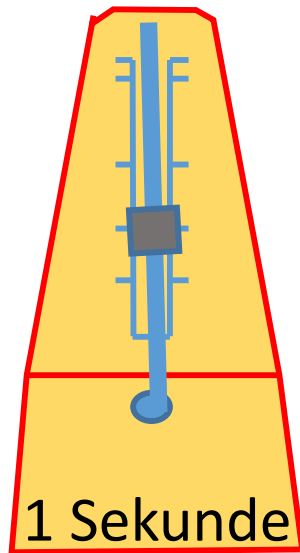


Timer



100

4



Licht

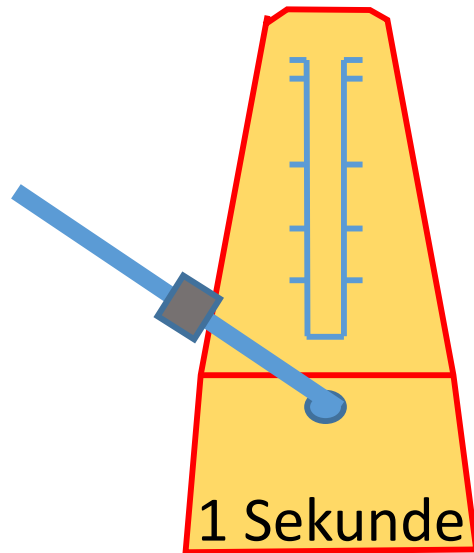


Timer



100

4



Licht

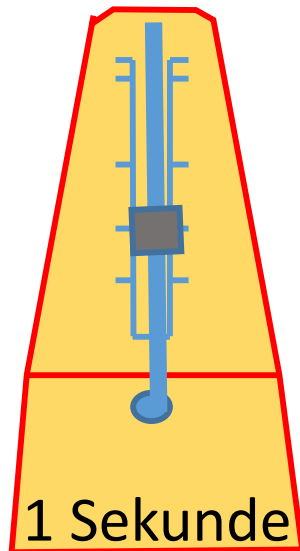


Timer



100

4



Licht

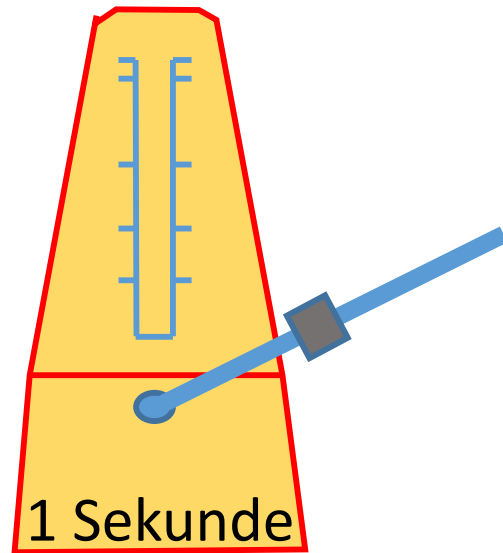


Timer



100

4



Licht

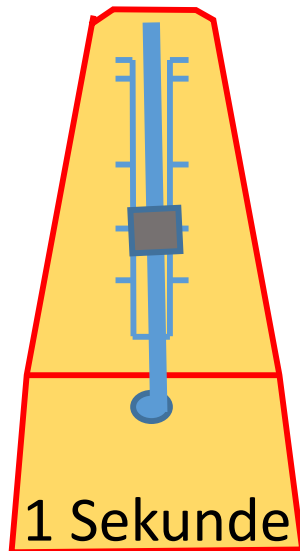


Timer

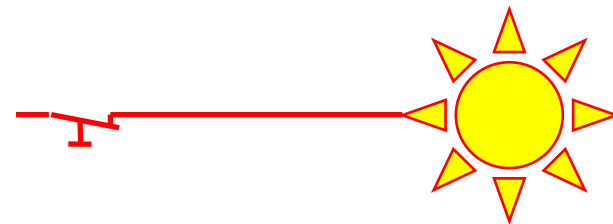


100

5



Licht

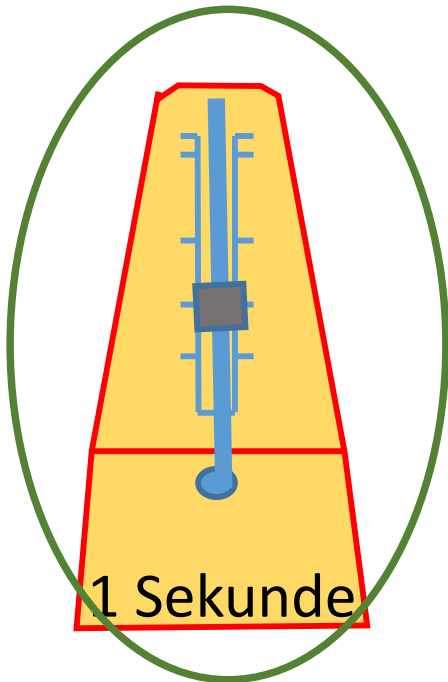


Timer

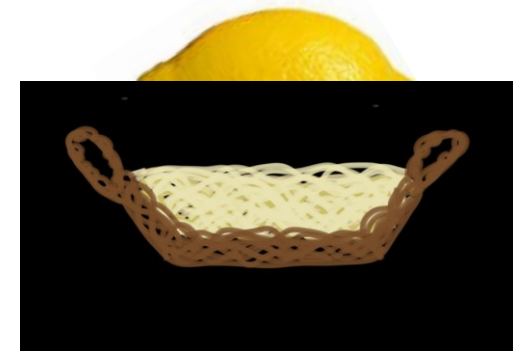


100

5



Licht

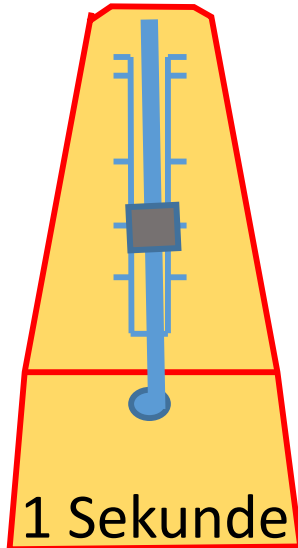
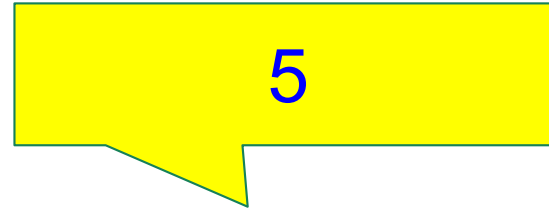
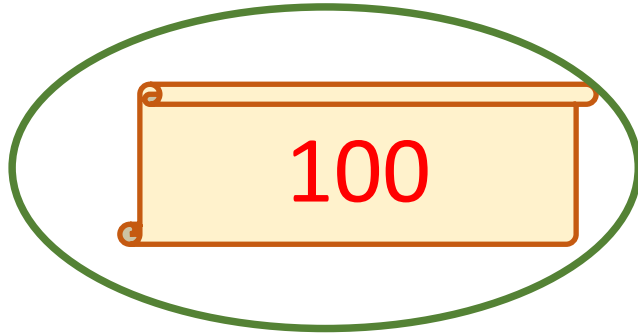


Prescaler PSC => Zählgeschwindigkeit



Timer

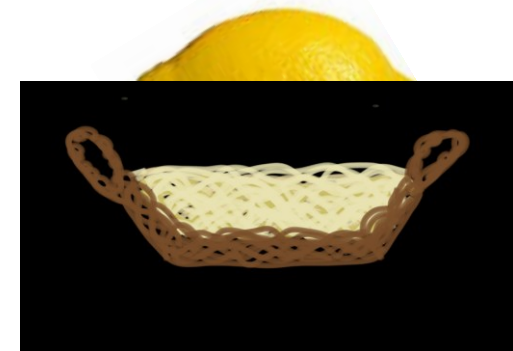
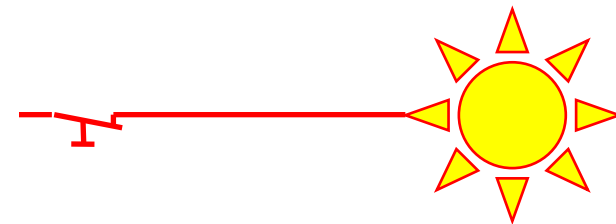
Autoreload Register ARR: So weit zählen



Prescaler PSC



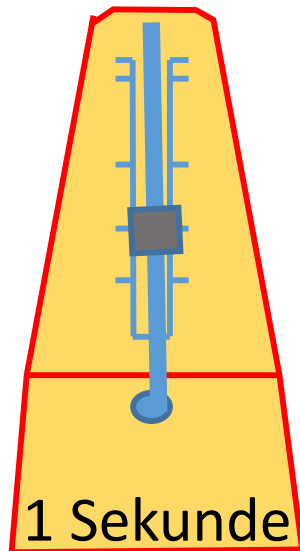
Licht



Timer

Autoreload Register ARR

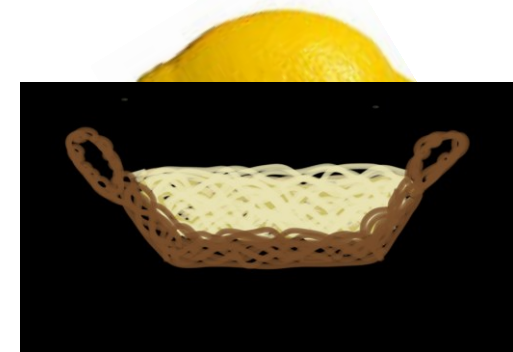
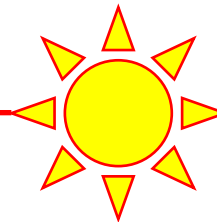
Counter CNT: Der Zähler



Prescaler PSC



Licht

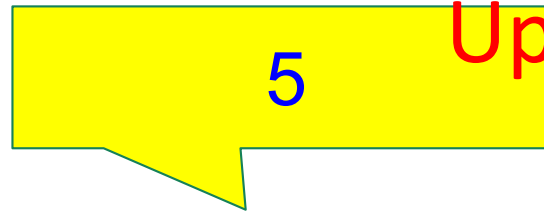


Timer

Autoreload Register ARR

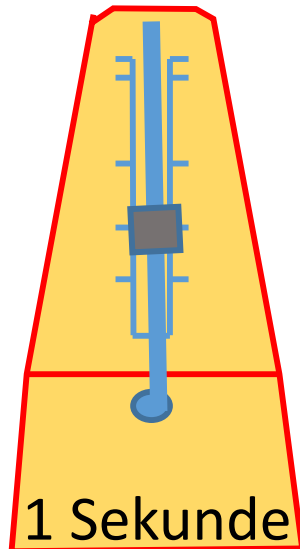


Counter CNT

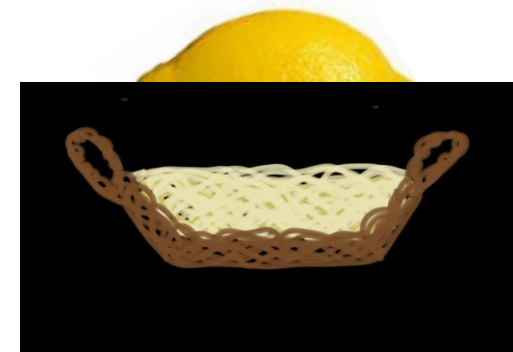
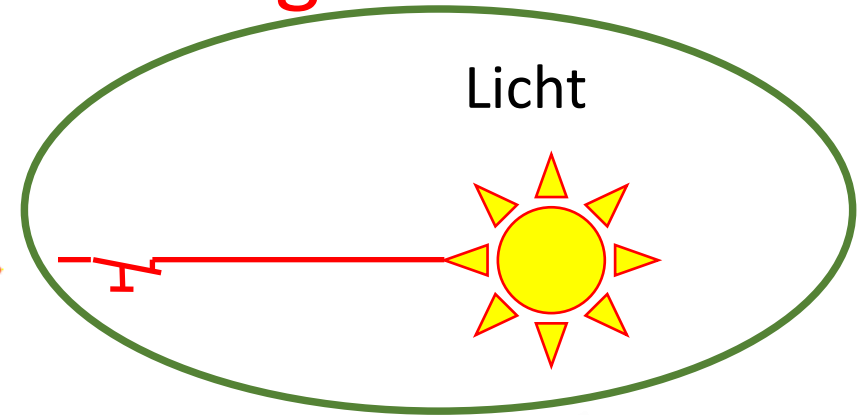


Update Interrupt Flag UIF

Zeigt Überlauf an



Prescaler PSC



Timer

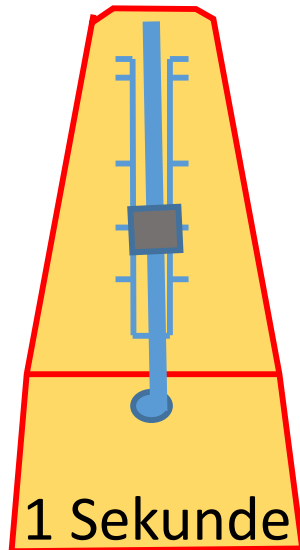
Autoreload Register ARR

100

Counter CNT

5

Update Interrupt Flag UIF
Licht



Prescaler PSC



Counter Enable CEN:



Schaltet zählen ein

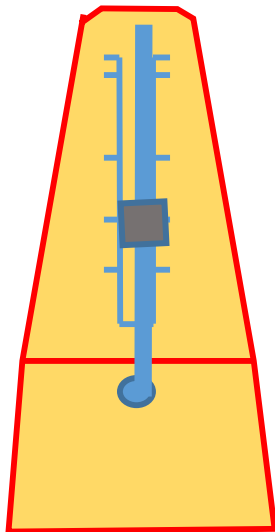


Timer

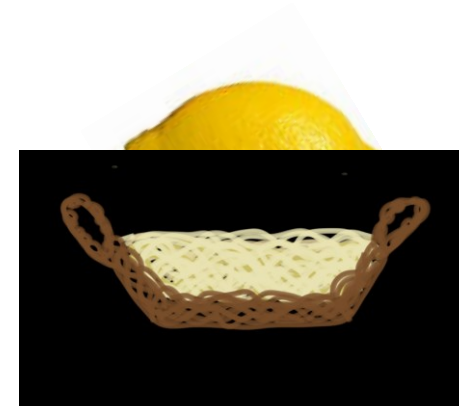
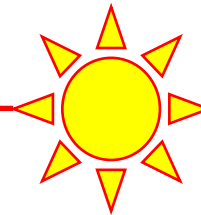


100

5



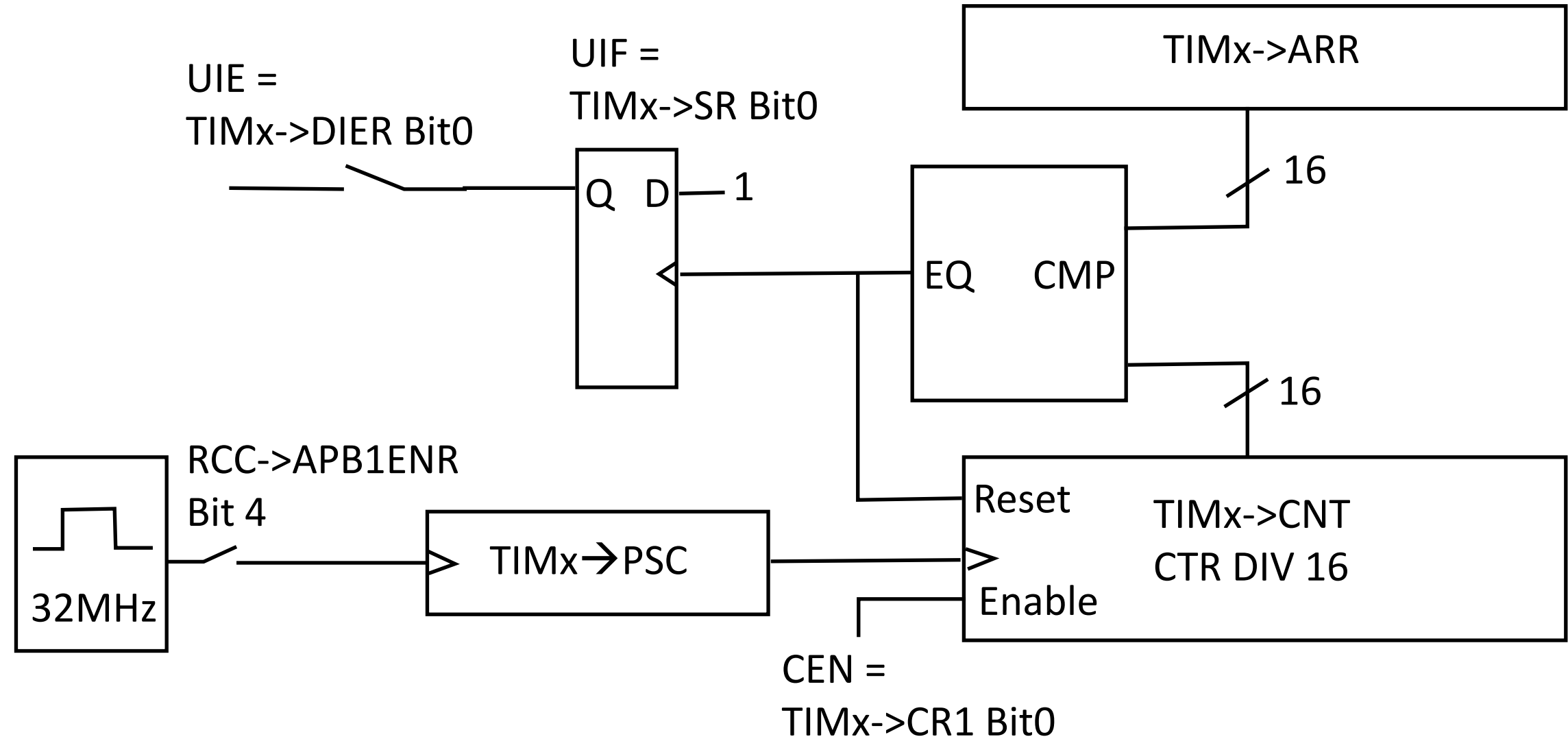
Licht



Basic Timer TIM6 oder TIM7



Timer

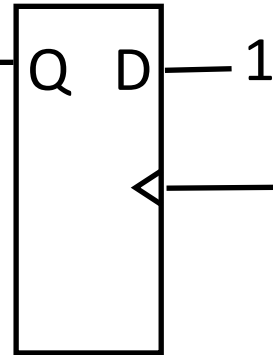


Timer



UIE =
TIMx->DIER Bit0

UIF =
TIMx->SR Bit0



16-Bit Binärzähler
Zählt von 0 bis höchstens
65535

TIMx->ARR

16

Reset

TIMx->CNT
CTR DIV 16

Enable

CEN =
TIMx->CR1 Bit0

RCC->APB1ENR
Bit 4

TIMx->PSC

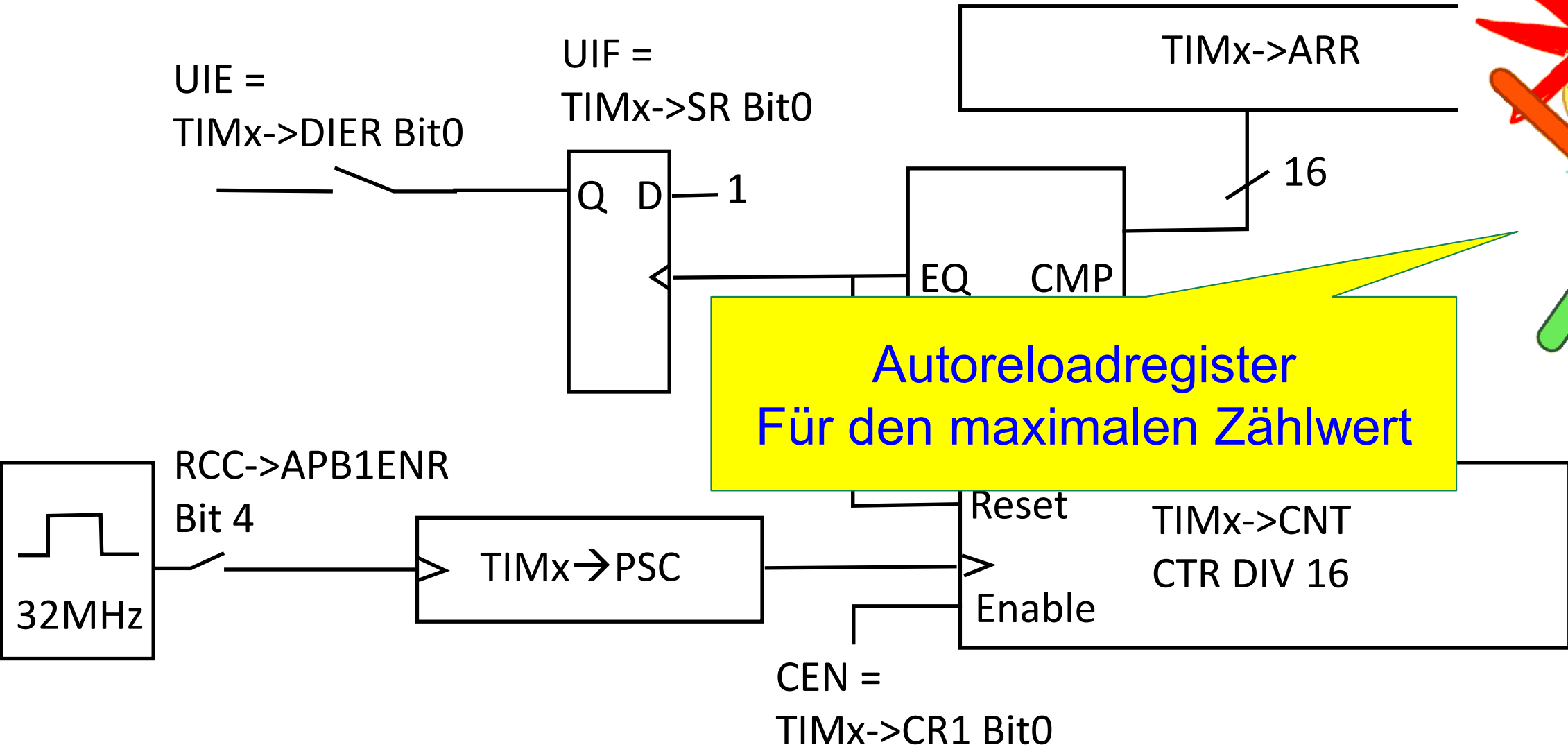
32MHz



Timer



Autoreloadregister
Für den maximalen Zählwert



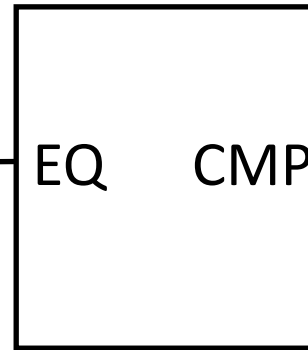
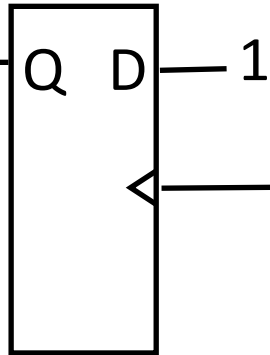
Timer



UIE =
TIMx->DIER Bit0

UIF =
TIMx->SR Bit0

TIMx->ARR



RCC->APB1ENR
Bit 4

32MHz

TIMx->PSC

Reset

CEN =
TIMx->

Vergleichen
Wenn ARR=CNT dann von
vorne

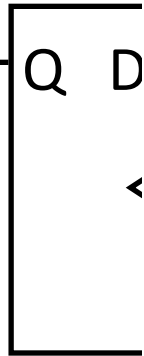


Timer



UIE =
TIMx->DIER Bit0

UIF =
TIMx->



TIMx->ARR

16

Der Überlauf wird im UIF
Update Interrupt Flag
angezeigt

RCC->APB1ENR
Bit 4

32MHz

TIMx->PSC

Reset

TIMx->CNT
CTR DIV 16

Enable

CEN =
TIMx->CR1 Bit0

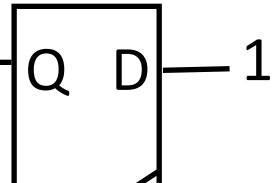


Timer

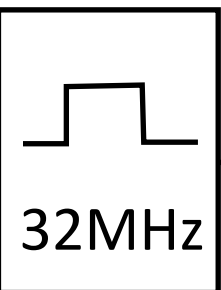
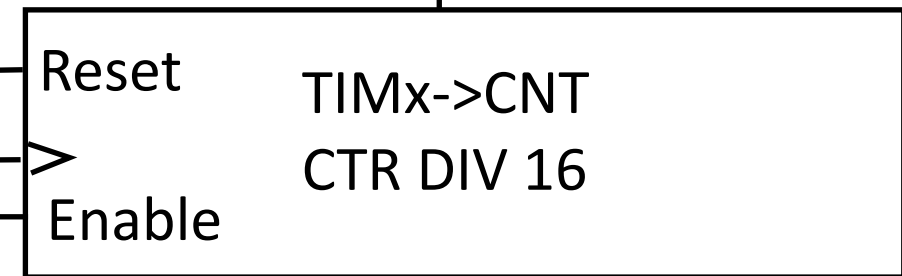
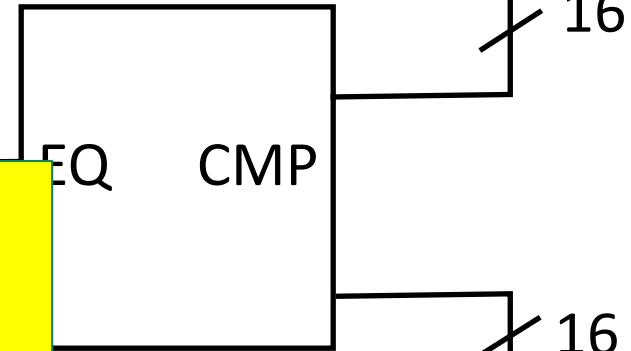
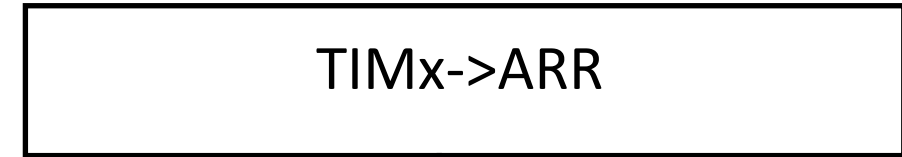


UIE =
TIMx->DIER Bit0

UIF =
TIMx->SR Bit0



Mit Counter Enable
CEN=1 wird der Counter
freigegeben



RCC->APB1ENR
Bit 4



EN =
TIMx->CR1 Bit0



Timer



UIE =
TIMx->DIER Bit0

UIF =
TIMx->SR Bit0

Der Prescaler stellt die
Zählgeschwindigkeit ein:
 $31 = 1\mu s$
 $31999 = 1ms$

TIMx->ARR

EQ CMP

16

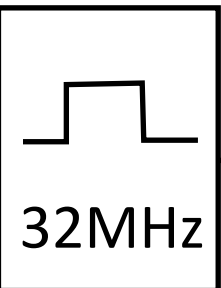
16

Reset

TIMx->CNT
CTR DIV 16

Enable

CEN =
TIMx->CR1 Bit0



TIMx->PSC



Timer



UIE =
TIMx->D

Der Timer muss mit 32MHz
Takt versorgt werden um
arbeiten zu können

TIMx->ARR

16

MP

16

RCC->APB1ENR

Bit

32MHz

TIMx->PSC

Reset

TIMx->CNT
CTR DIV 16

Enable

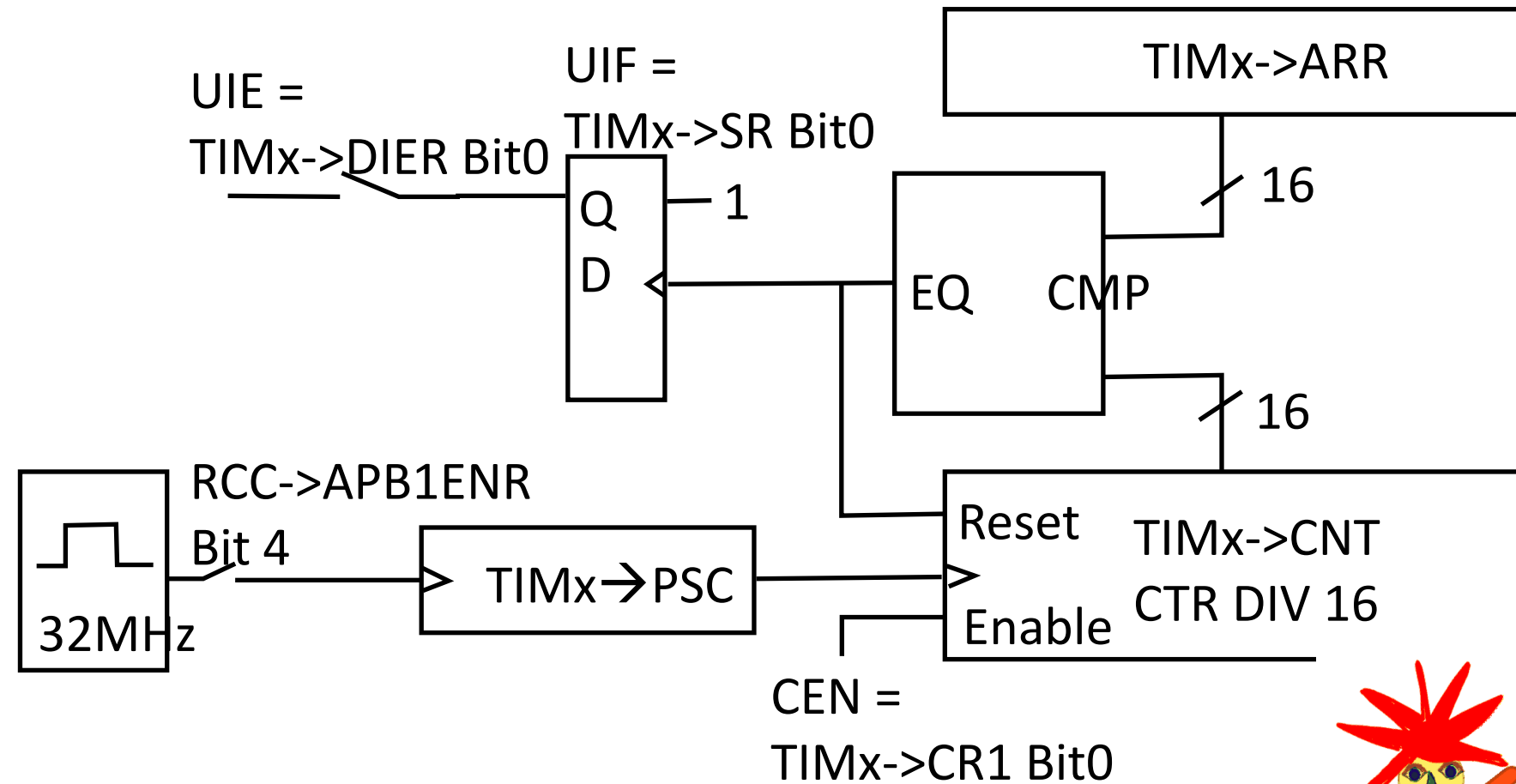
CEN =
TIMx->CR1 Bit0



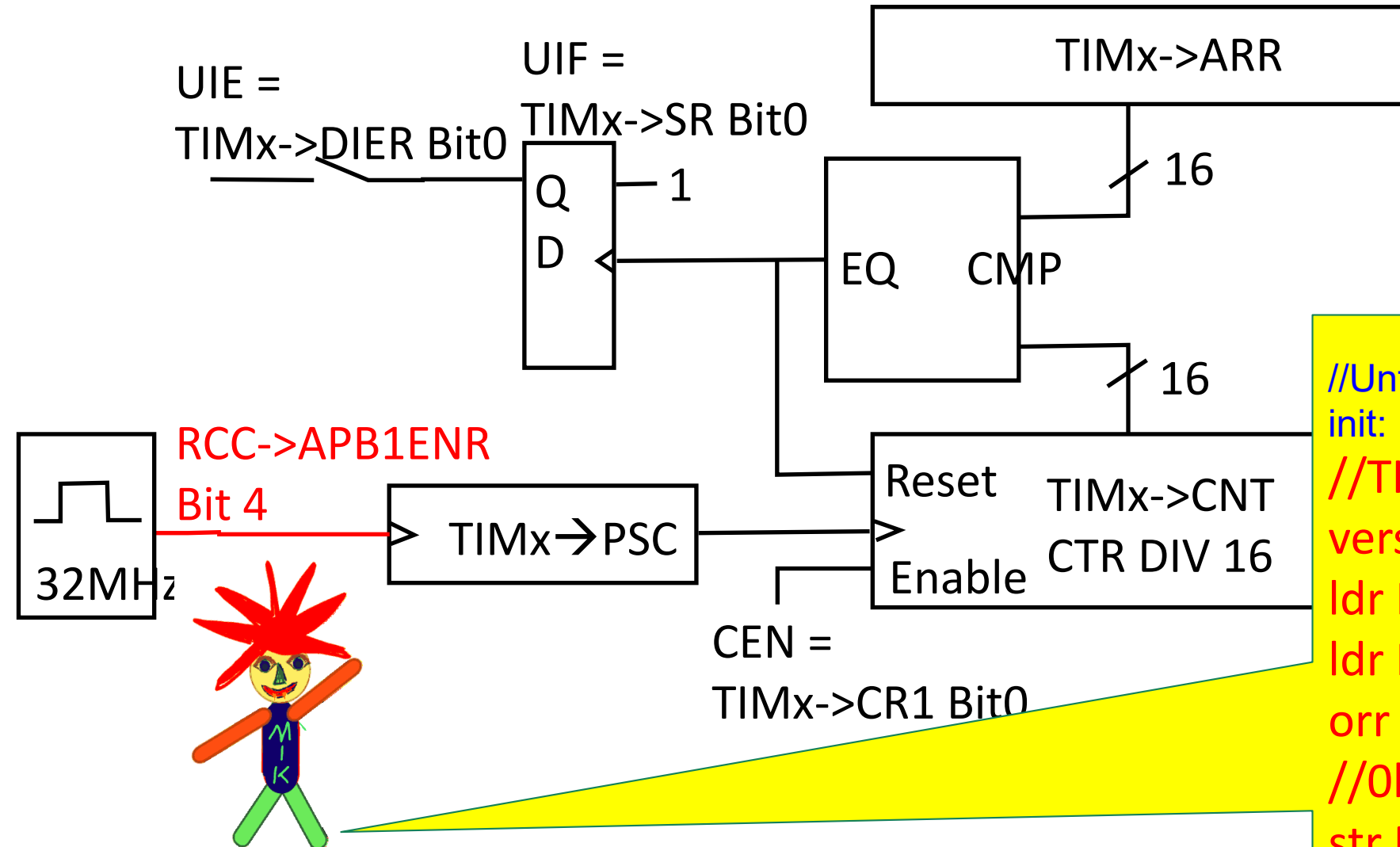
Timer



Beispiel: Eine LED
soll mit 1Hz blinken



Timer



//Unterprogramm
init:

//TIM2, TIM6 und TIM7 mit Takt
versorgen

ldr R0,=rcc

ldr R1,[R0,RCC_APB1ENR]

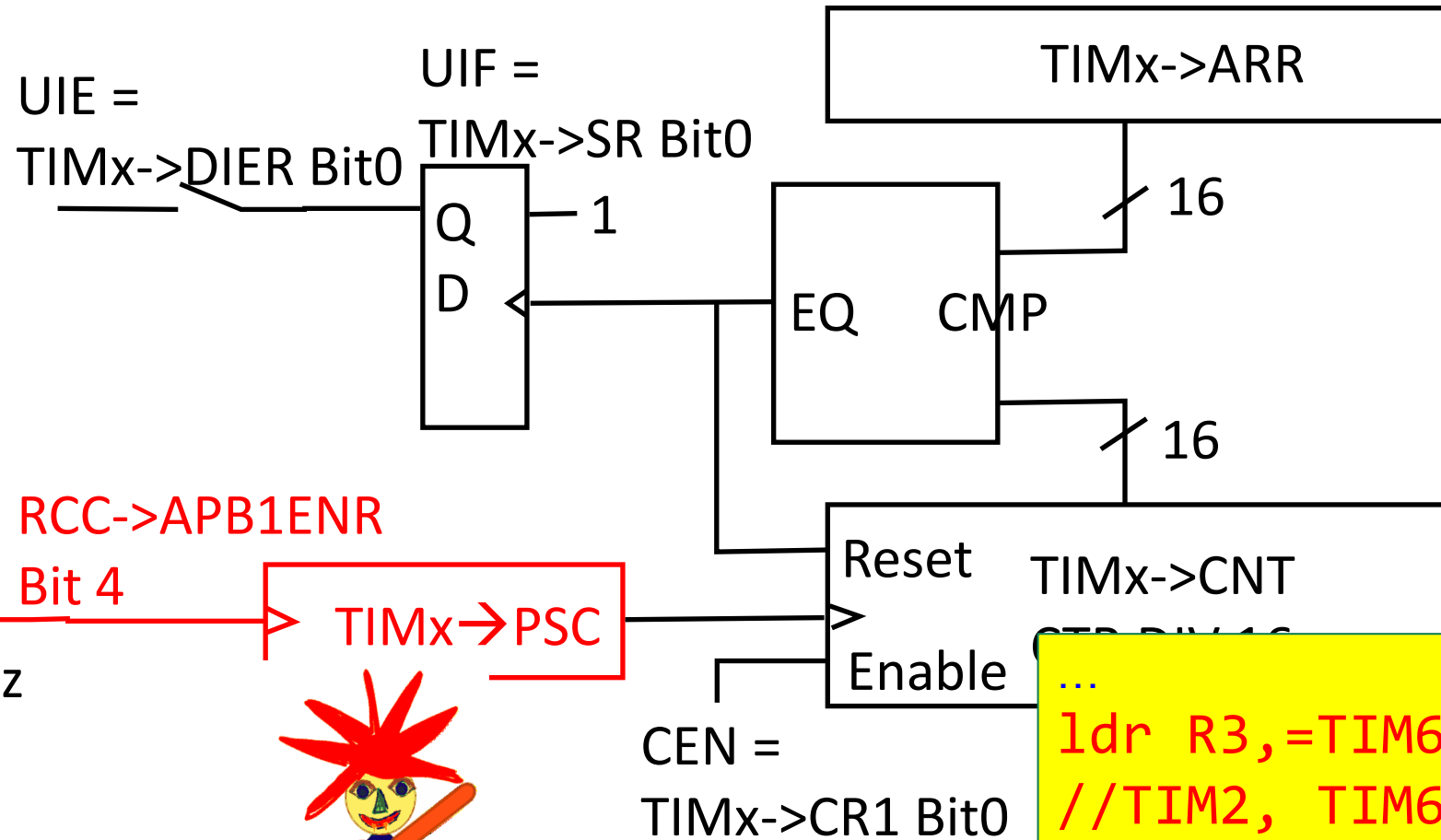
orr R1,0b110001

//0bTIM7,TIM6,000,TIM2

str R1,[R0,RCC_APB1ENR]



Timer



```
...
ldr R3,=TIM6
//TIM2, TIM6, TIM7
mov R1,#31999 //Prescalerwert
31999=1ms
str R1,[R3,PSC]
```



Timer

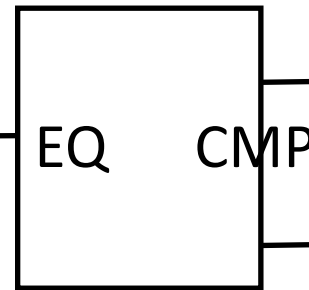
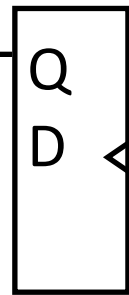


```
...  
mov R1,#499  
//Autoreloadwert 500ms  
str R1,[R3,ARR]
```

TIMx->ARR

UIE =
TIMx->DIER Bit0

UIF =
TIMx->SR Bit0



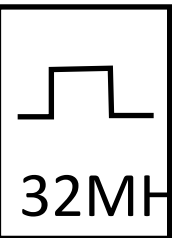
16

16

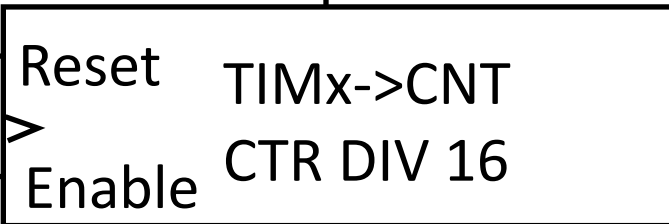
RCC->APB1ENR

Bit 4

TIMx->PSC



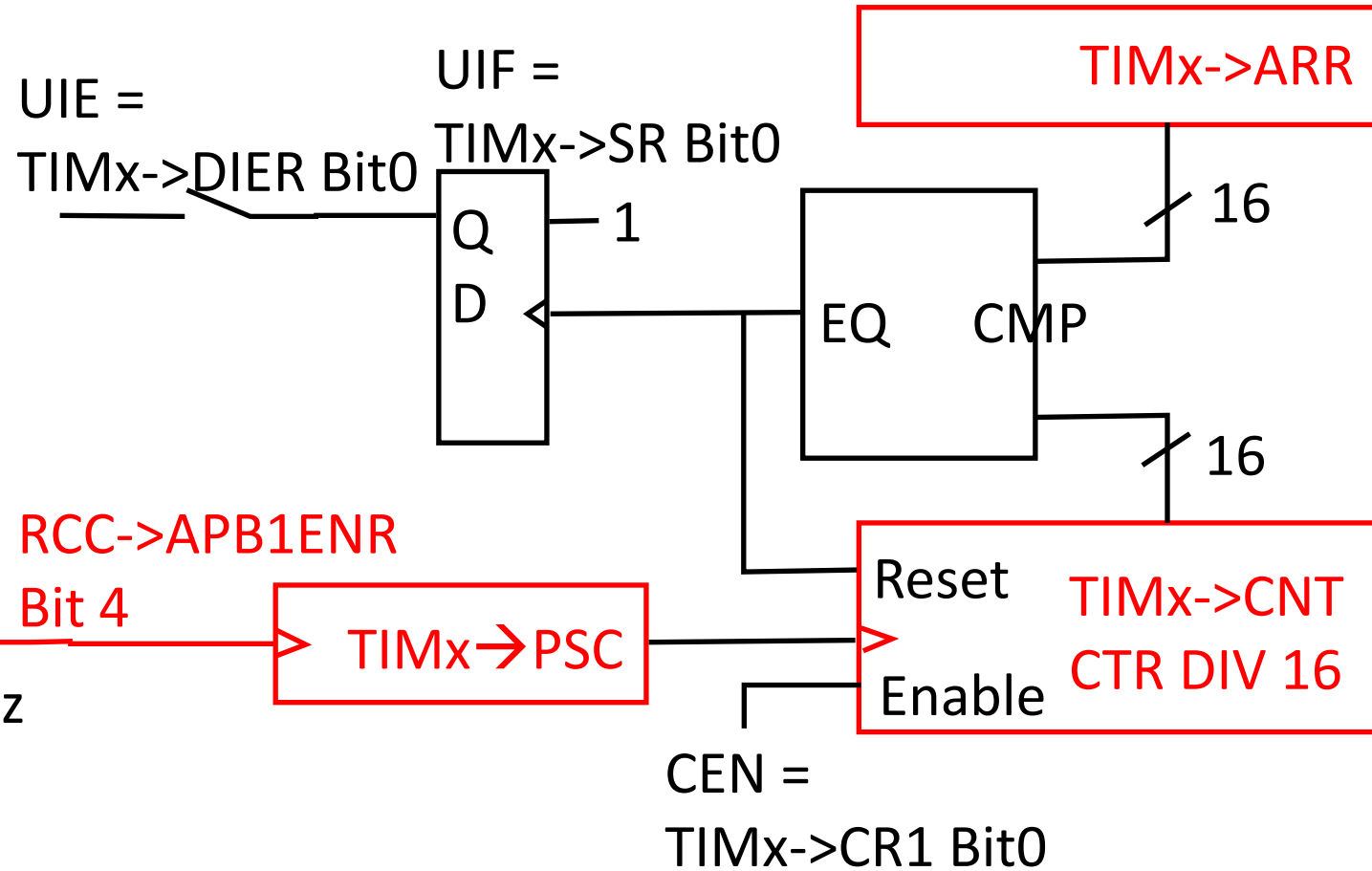
32MHz



CEN =
TIMx->CR1 Bit0



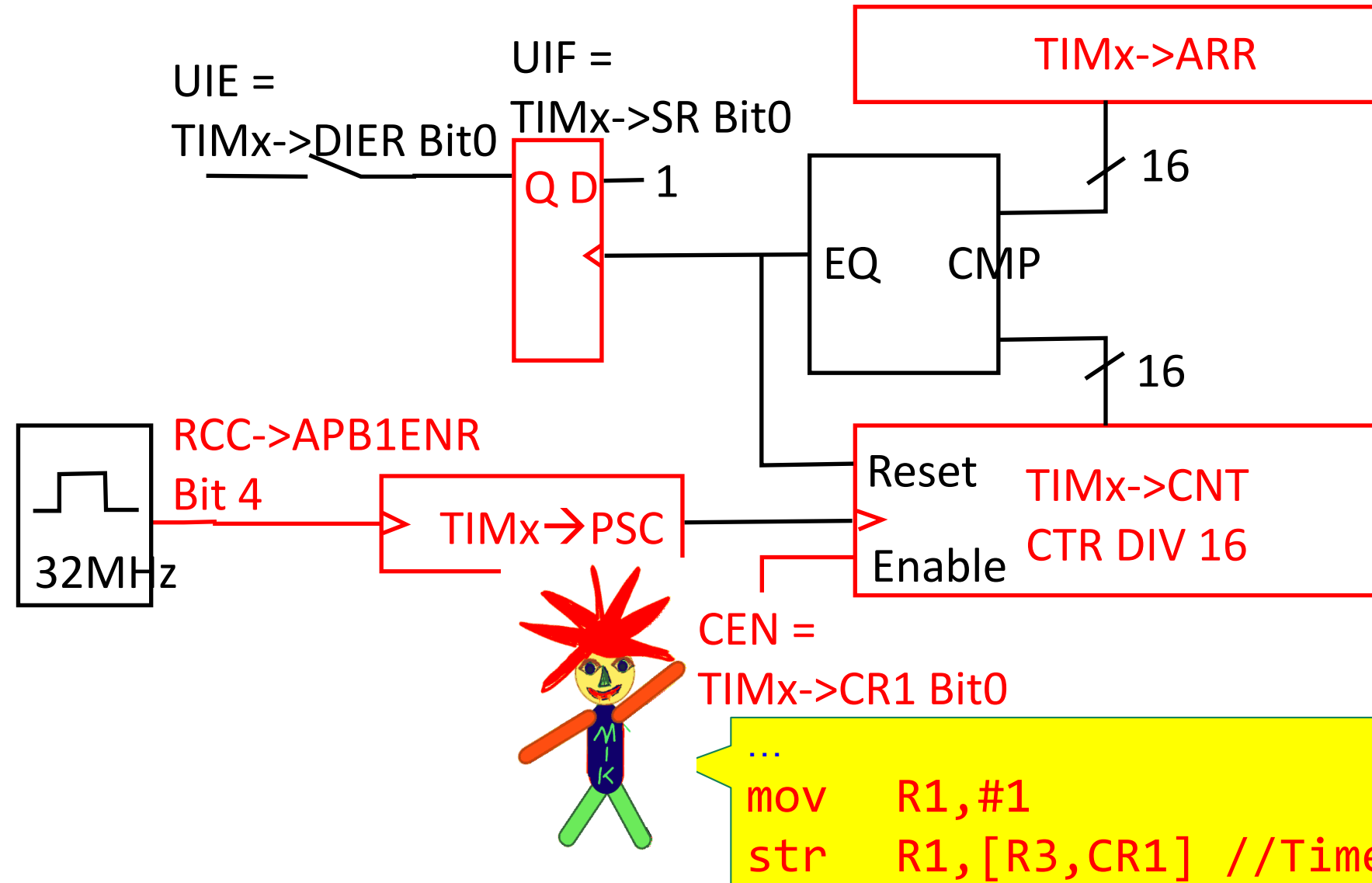
Timer



```
...  
mov R1,#0  
str R1,[R3,CNT]  
//Zähler bei 0 starten
```



Timer



Timer

UIE =
TIMx-



Die
Endlos-
schleife

32MHz

```
ldr    R1,=GPIOC
```

```
schleife://Sprungmarke
```

```
ldr    R2,[R3,SR] //Statusregister einlesen R3=TIM
```

```
tst    R2,Bit0
```

```
//Bit0 = Update Interrupt Flag UIF = Überlaufbit
```

```
beq    schleife    //nicht gesetzt dann Schleife
```

```
mov    R2,#0        //UIF zurücksetzen
```

```
str    R2,[R3,SR]
```

```
ldrb   R2,[R1,ODR]    //Bit1 im ODR umschalten
```

```
(blinken)
```

```
eor    R2,Bit1
```

```
strb   R2,[R1,ODR]
```

```
b      schleife
```



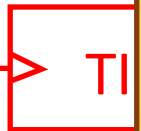
Timer

UIE =
TIMx-



Autoreload
prüfen

CB1ENR



```
ldr    R1,=GPIOC
schleife://Sprungmarke
ldr    R2,[R3,SR] //Statusregister einlesen R3=TIM
tst    R2,Bit0
//Bit0 = Update Interrupt Flag UIF = Überlaufbit
beq    schleife    //nicht gesetzt dann Schleife
mov    R2,#0        //UIF zurücksetzen
str    R2,[R3,SR]
ldrb   R2,[R1,ODR]   //Bit1 im ODR umschalten
(blinken)
eor    R2,Bit1
strb   R2,[R1,ODR]
b      schleife
```



Timer

Update
Interrupt
Flag
rücksetzen

ER Bit



RCC-
Bit 4

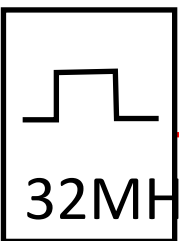
32MHz

```
ldr    R1,=GPIOC
schleife://Sprungmarke
ldr    R2,[R3,SR] //Statusregister einlesen R3=TIM
tst     R2,Bit0
//Bit0 = Update Interrupt Flag UIF = Überlaufbit
beq     schleife //nicht gesetzt dann Schleife
mov     R2,#0      //UIF zurücksetzen
str     R2,[R3,SR]
ldrb    R2,[R1,ODR]//Bit1 im ODR umschalten (blinken)
eor     R2,Bit1
strb    R2,[R1,ODR]
b       schleife
```



Timer

LED PC0
umschalten
mit XOR
(eor)



RCC->APB1ENR

Bit



TI

```
ldr    R1,=GPIOC
schleife://Sprungmarke
ldr    R2,[R3,SR] //Statusregister einlesen R3=TIM
tst    R2,Bit0
//Bit0 = Update Interrupt Flag UIF = Überlaufbit
beq    schleife    //nicht gesetzt dann Schleife
mov    R2,#0        //UIF zurücksetzen
str    R2,[R3,SR]
ldrb   R2,[R1,ODR]//Bit1 im ODR umschalten (blinken)
eor    R2,Bit1
strb   R2,[R1,ODR]
b      schleife
```

