



Timer und Interrupt

Basic Timer TIM6 und TIM7



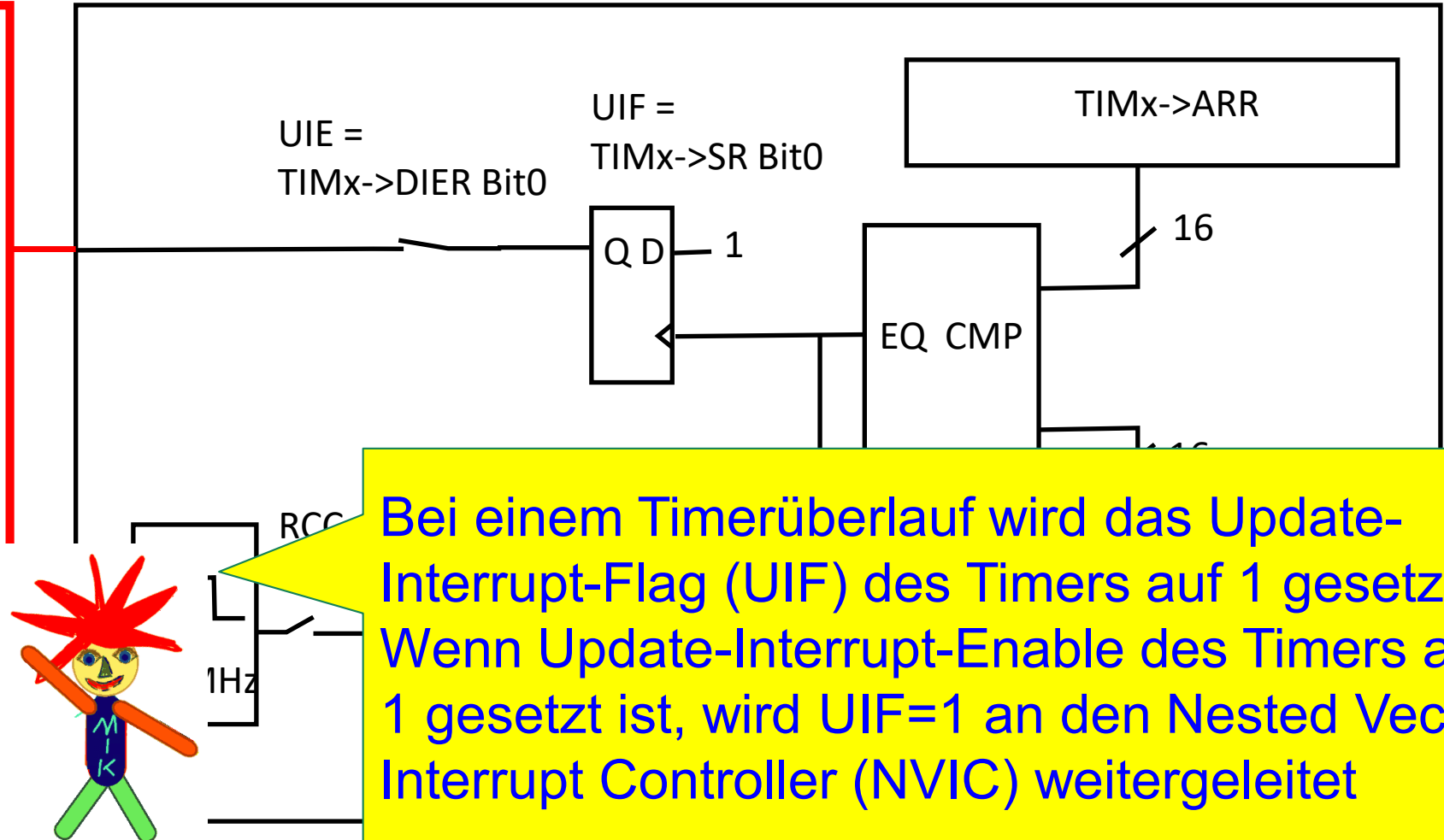
Ich bin Mik, Dein
Mikrocontroller



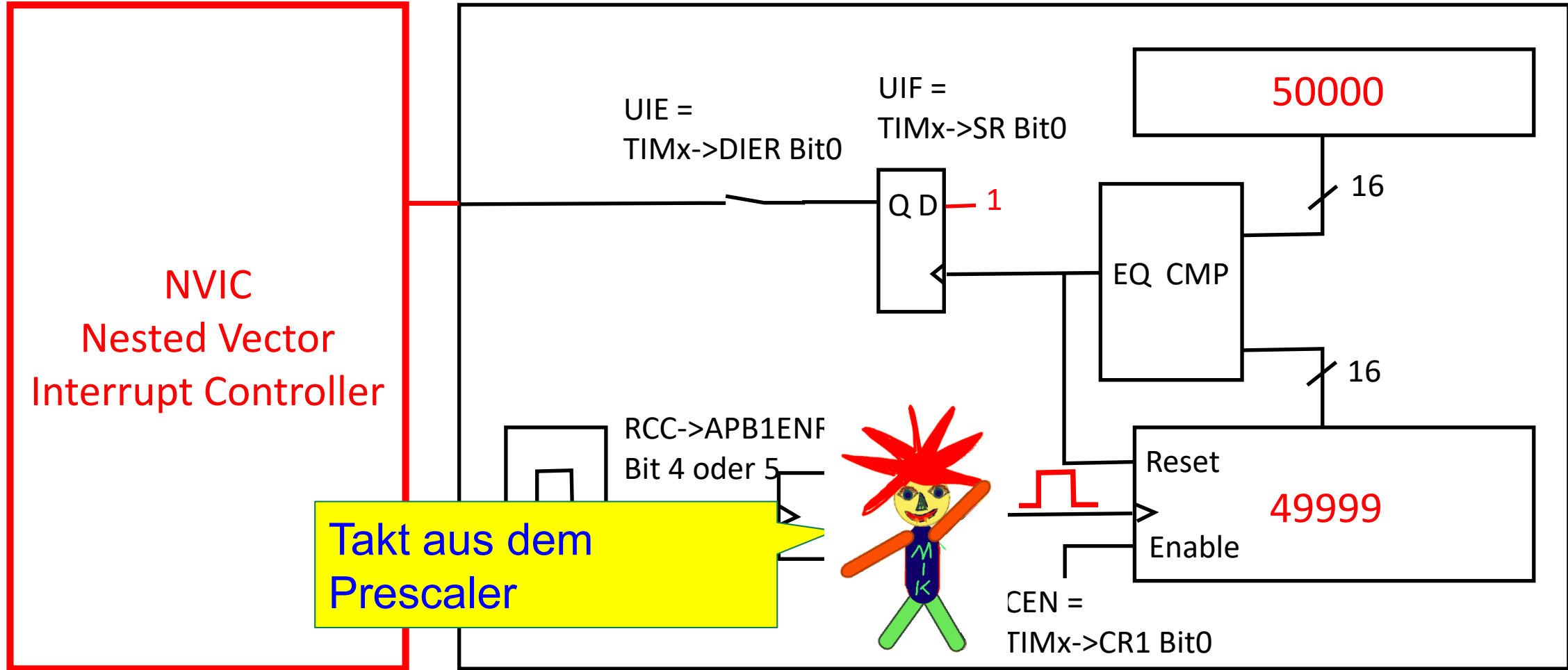
Timer und Interrupt



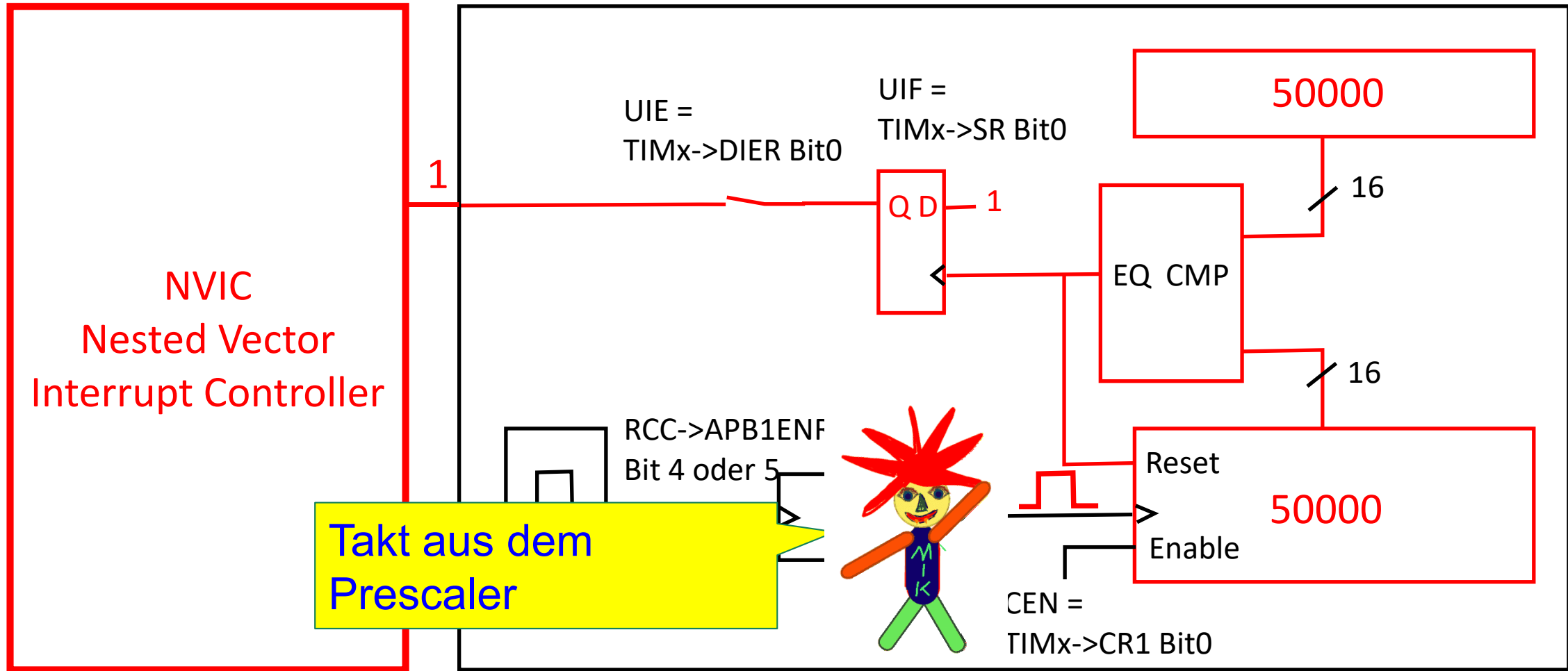
NVIC
Nested Vector
Interrupt Controller



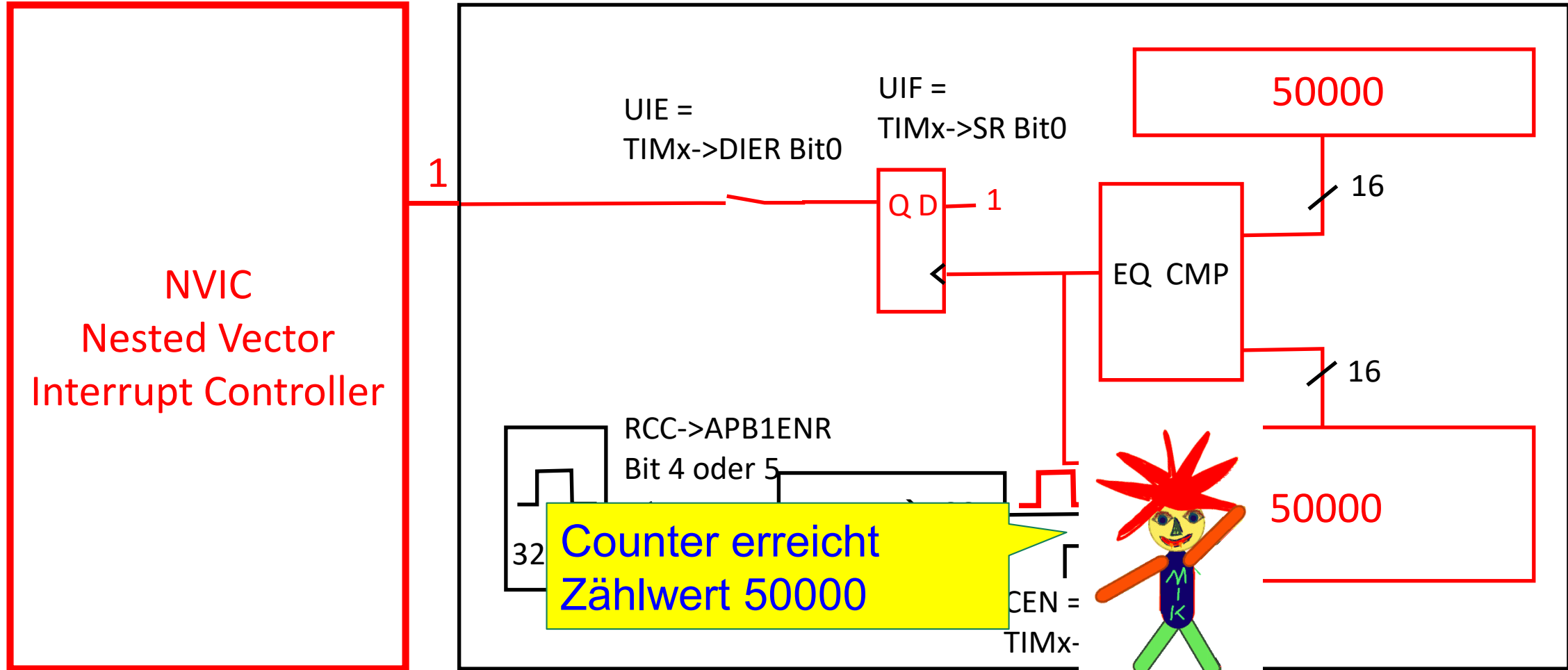
Timer und Interrupt



Timer und Interrupt



Timer und Interrupt

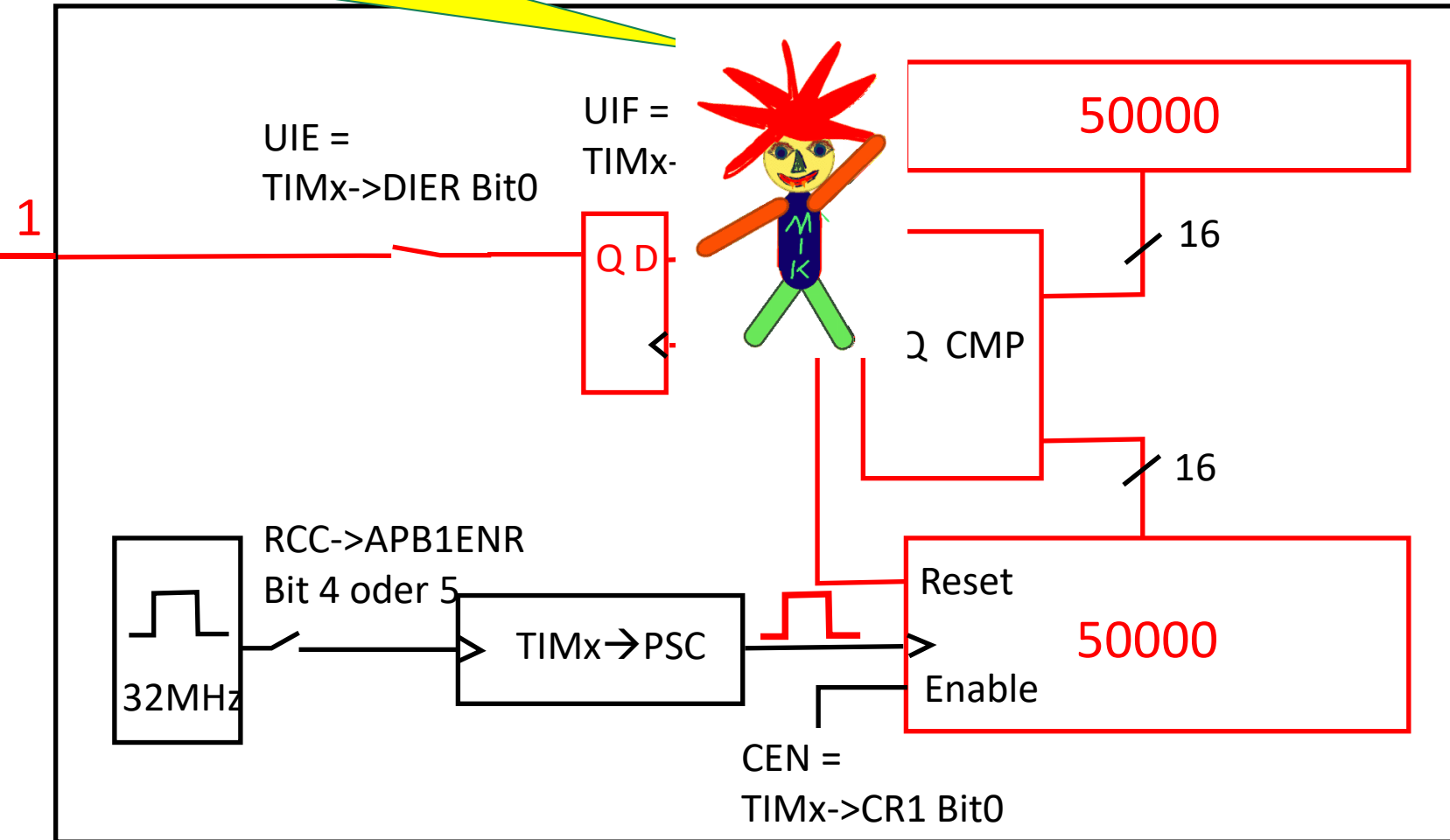


Timer und Interrupt



Der gleiche Wert steht im
Autoreloadregister: 50000

NVIC
Nested Vector
Interrupt Controller

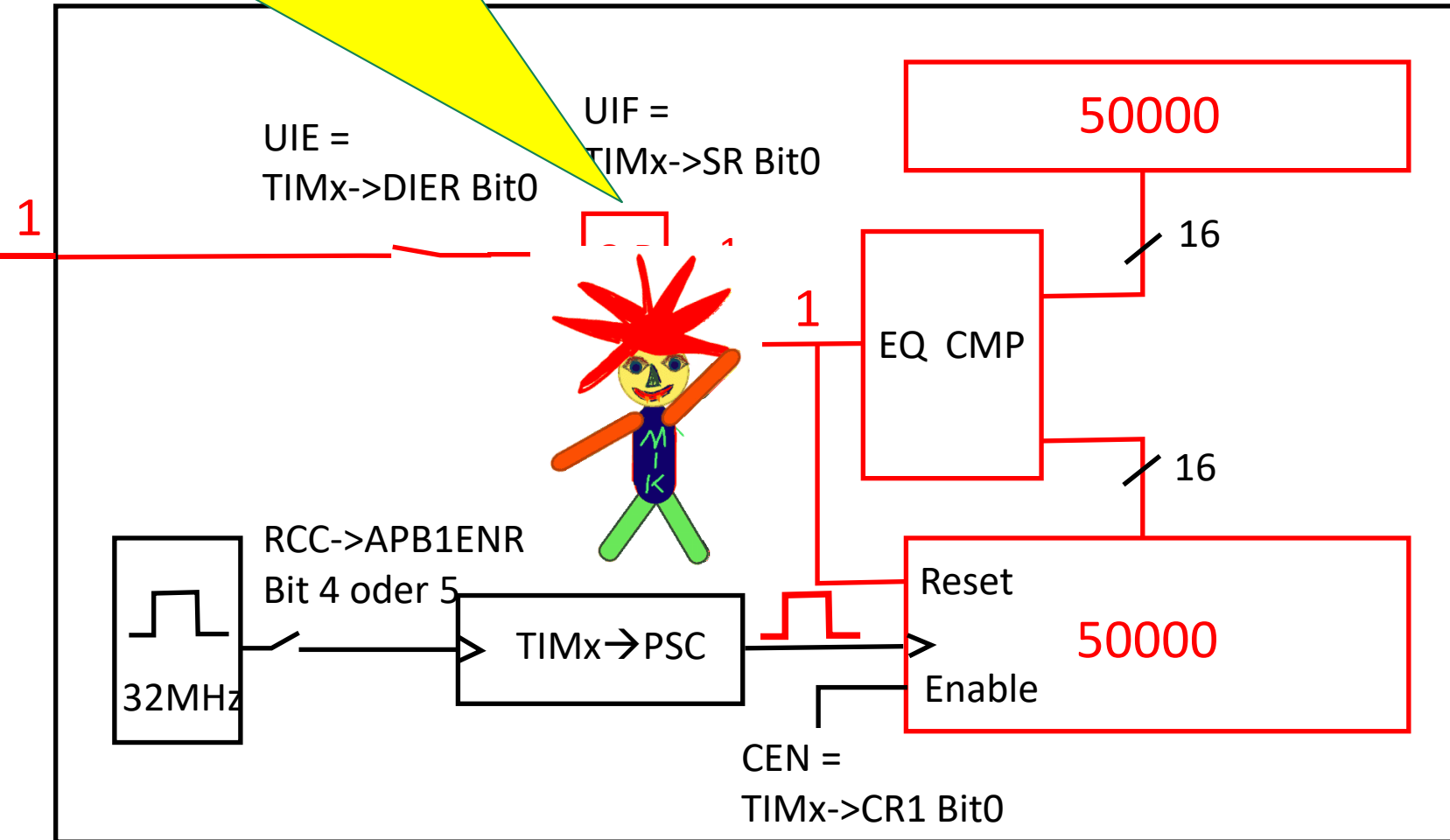


Timer und Interrupt

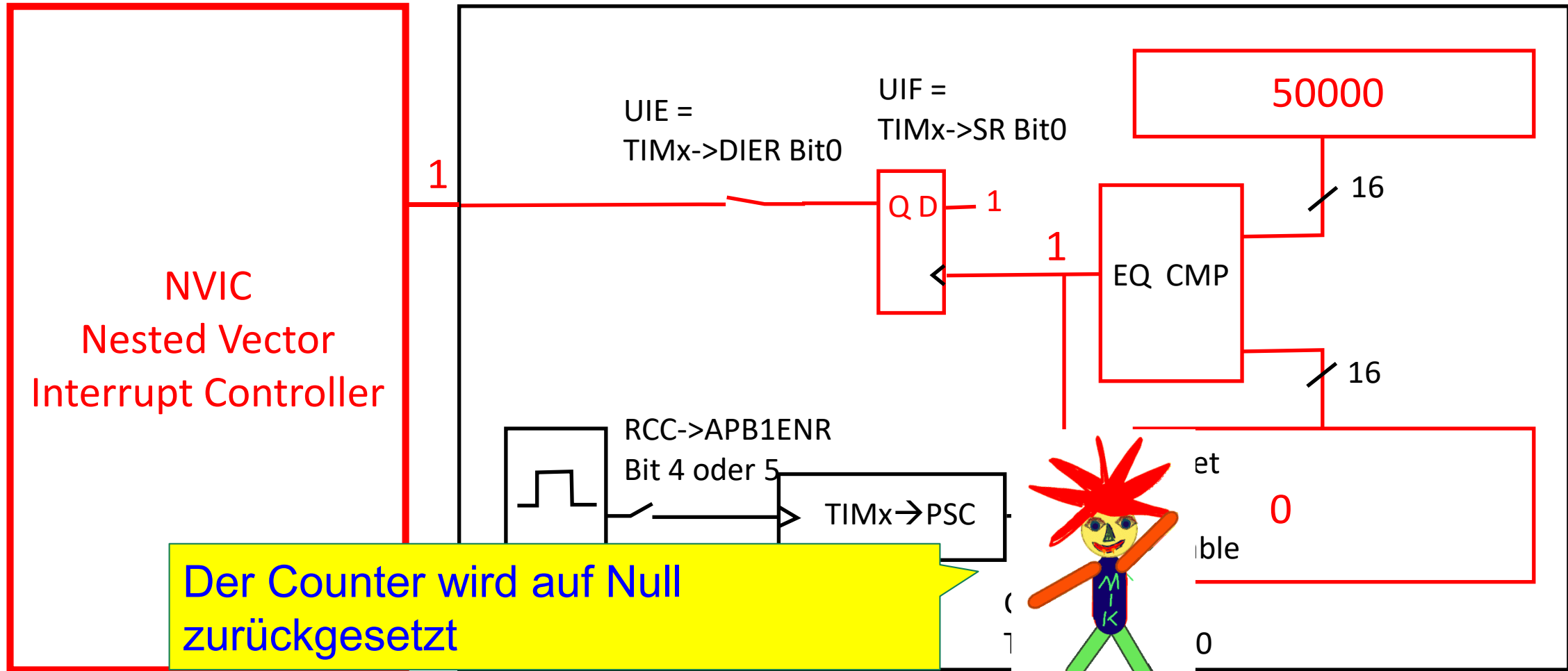


Der Vergleicherausgang wird 1

NVIC
Nested Vector
Interrupt Controller



Timer und Interrupt

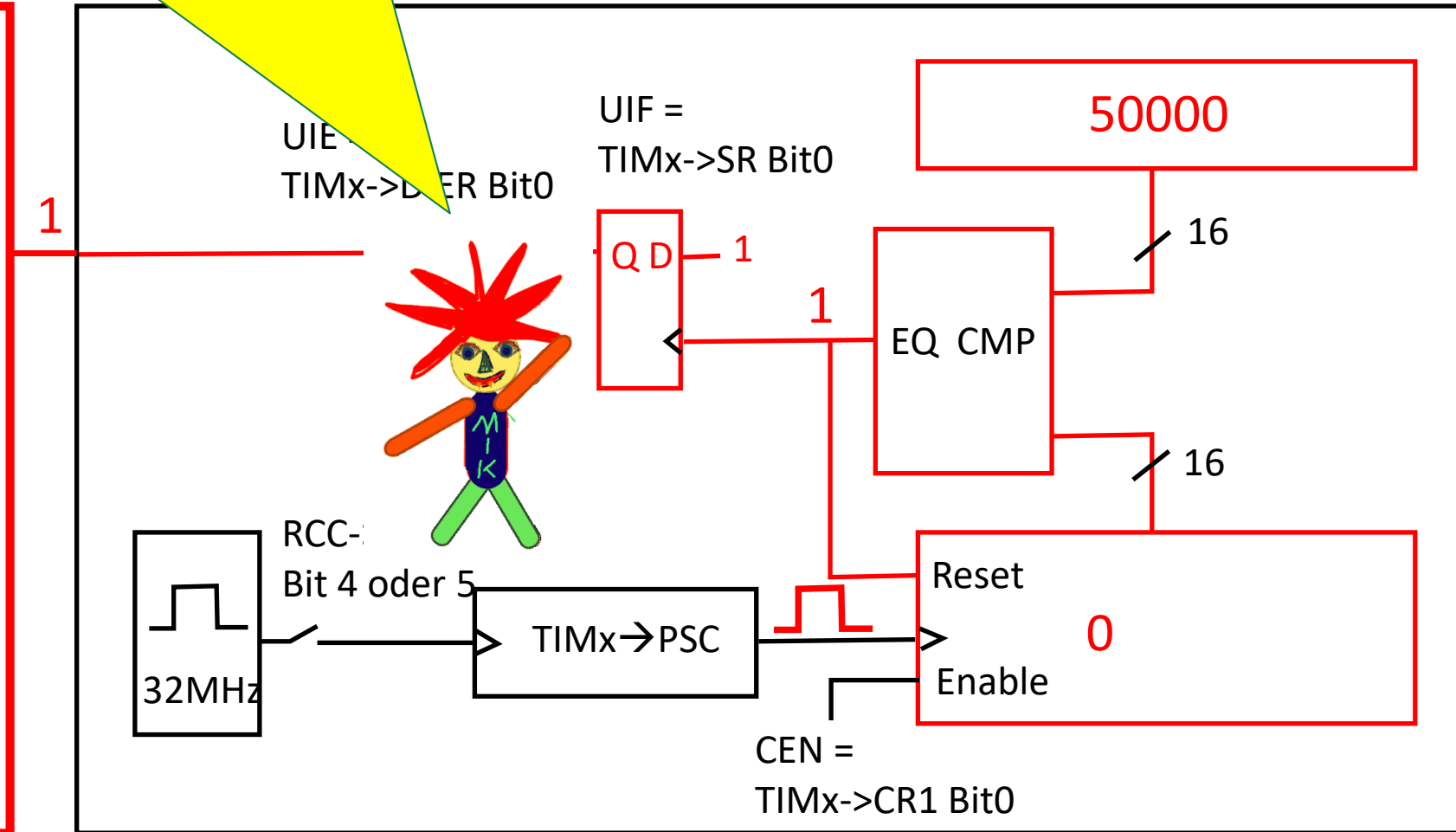


Timer und Interrupt



Gleichzeitig wird das UIF Flipflop (Update-Interrupt-Flag) auf 1 gesetzt

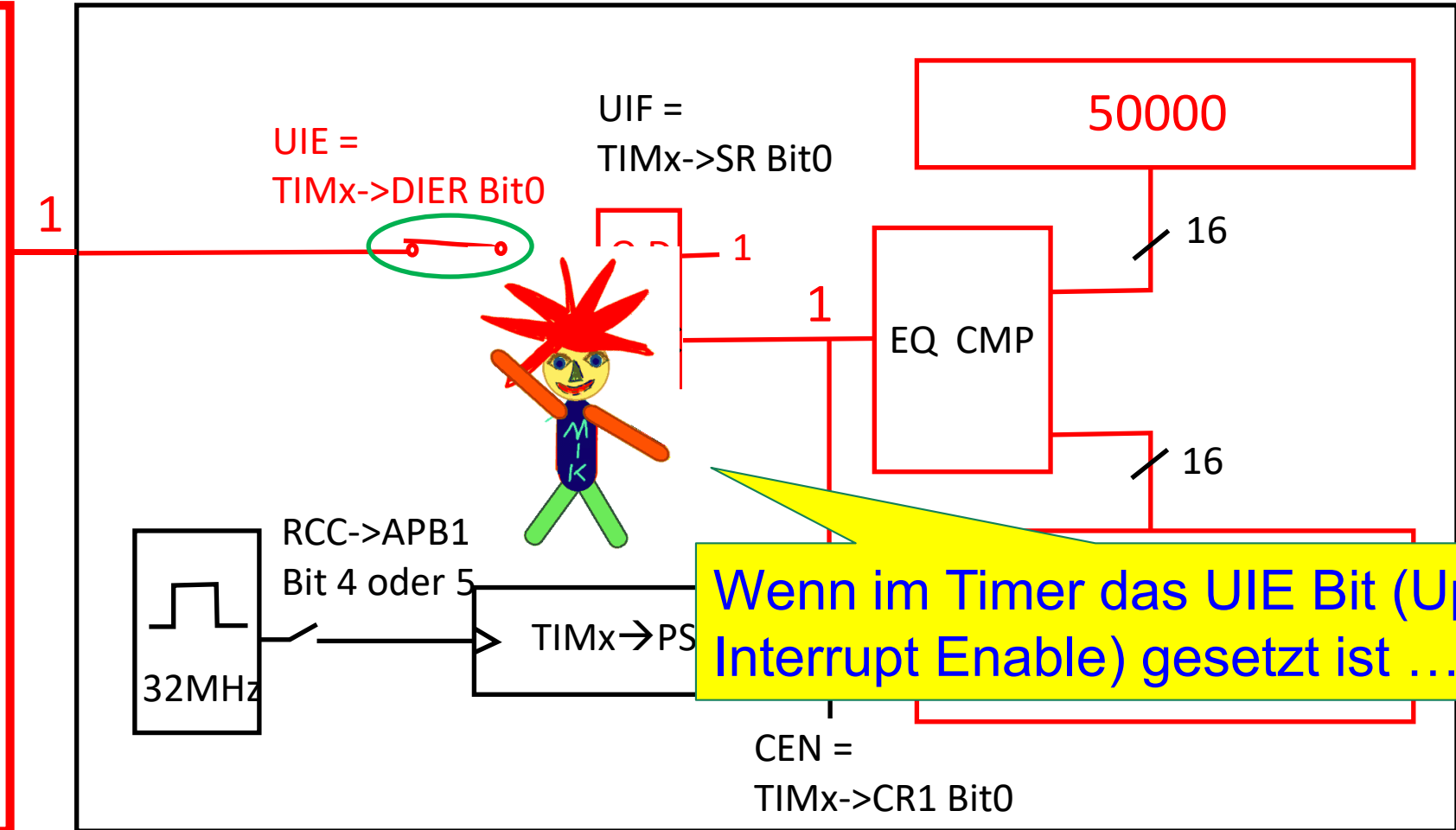
NVIC
Nested Vector
Interrupt Controller



Timer und Interrupt



NVIC
Nested Vector
Interrupt Controller

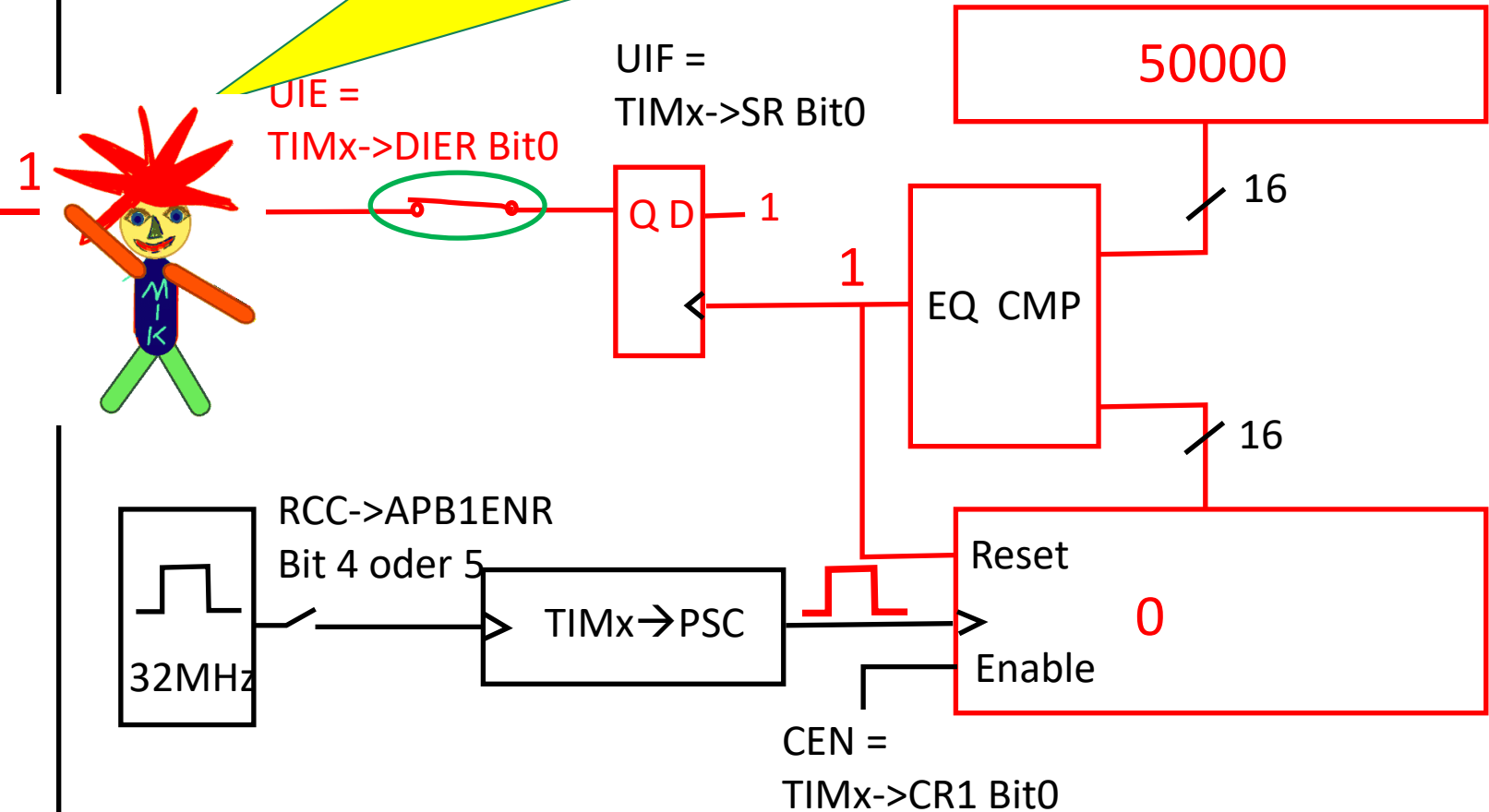


Timer und Interrupt



Wird die 1 an den Nested Vector Interrupt Controller (NVIC) zur Verarbeitung weitergeleitet

NVIC
Nested Vector
Interrupt Controller



A vintage-style alarm clock with a white face and Roman numerals, set against a light blue background. The clock has a dark metal frame and two bells on top. The time shown is approximately 10:10.

UIE = TIMx->DIER Bit0

50000

1

1

TIMx->DIER Bit0

— Q D — 1

1

EQ CMP

16

16

RCC->APB1ENR
Bit 4 oder 5

32MHz

TIMx → PSC

Reset

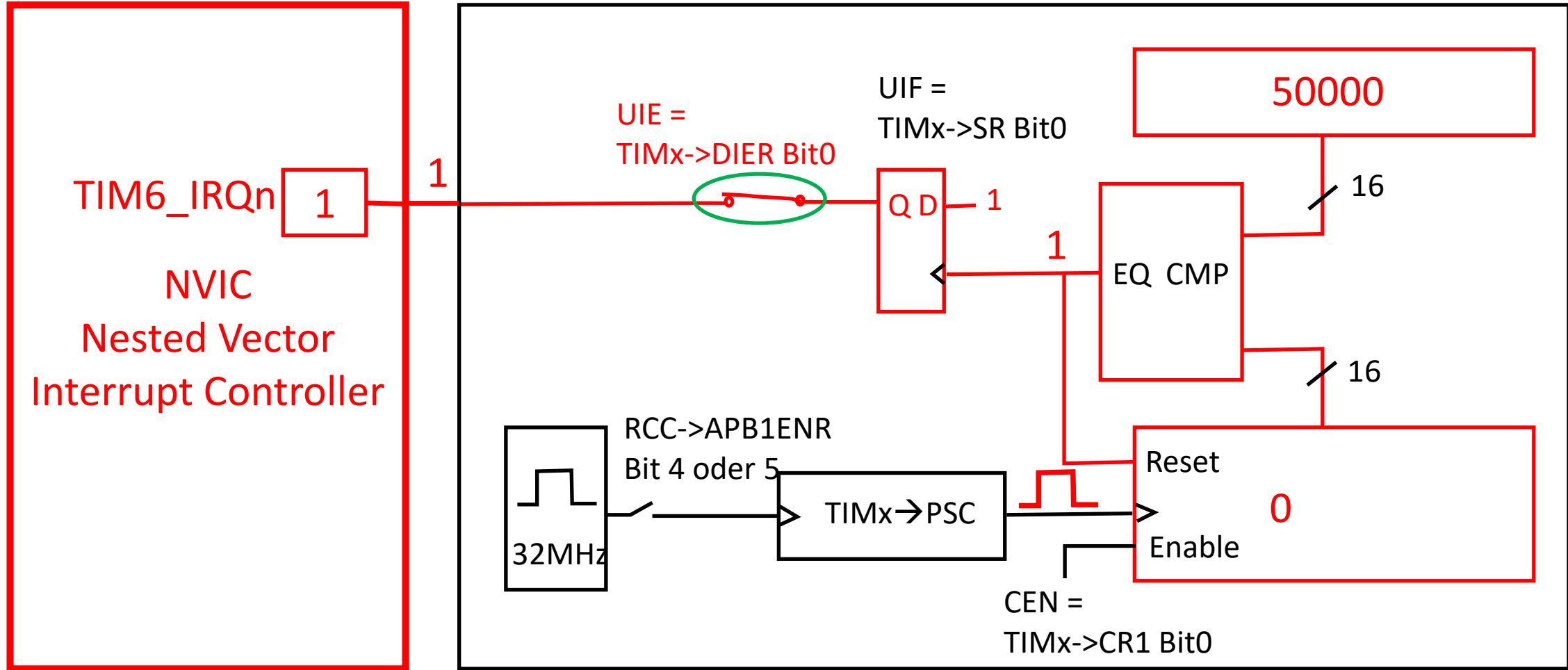
- Enable

0

CEN =
TIMx->CR1 Bit0

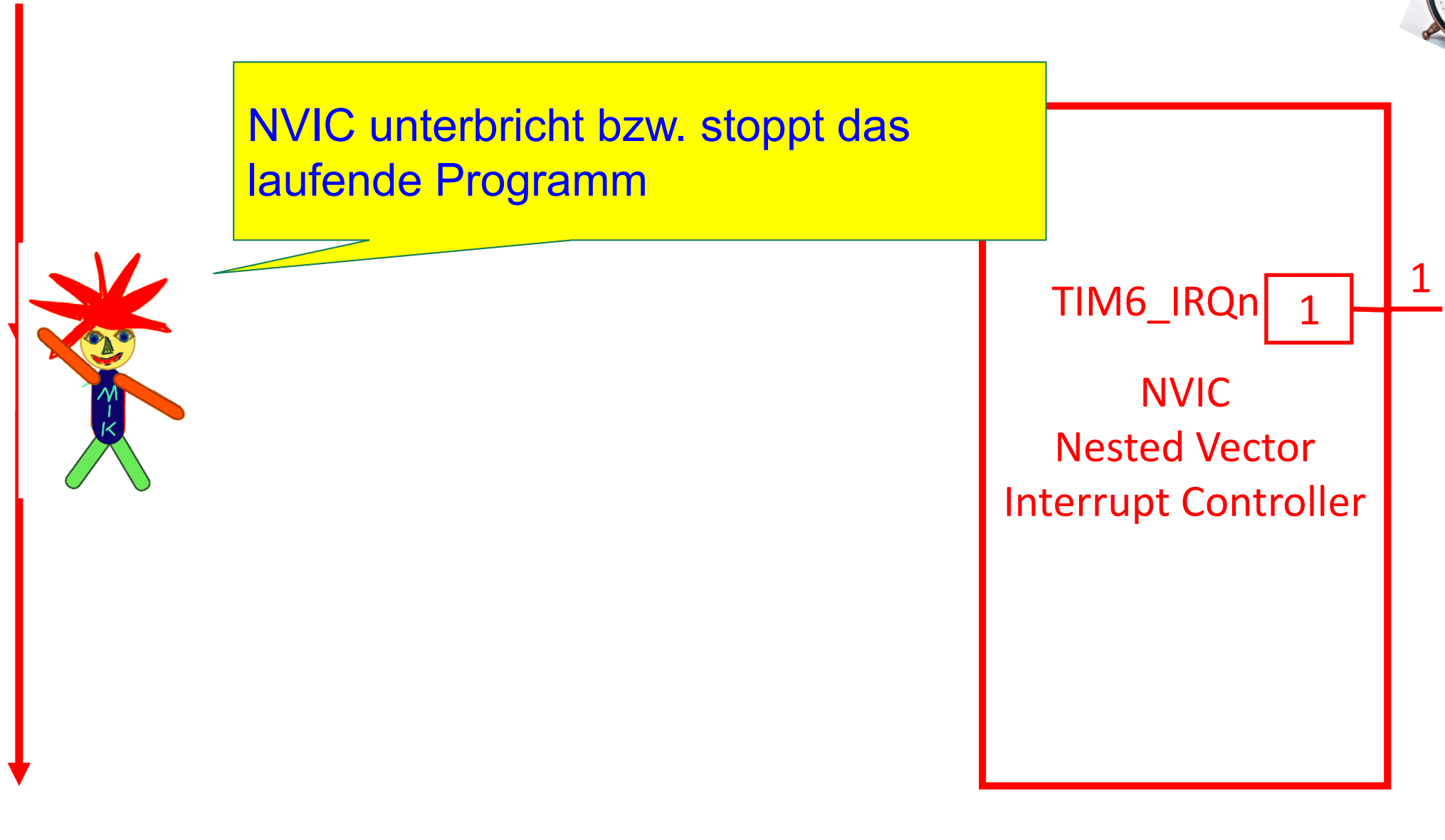


Timer und Interrupt



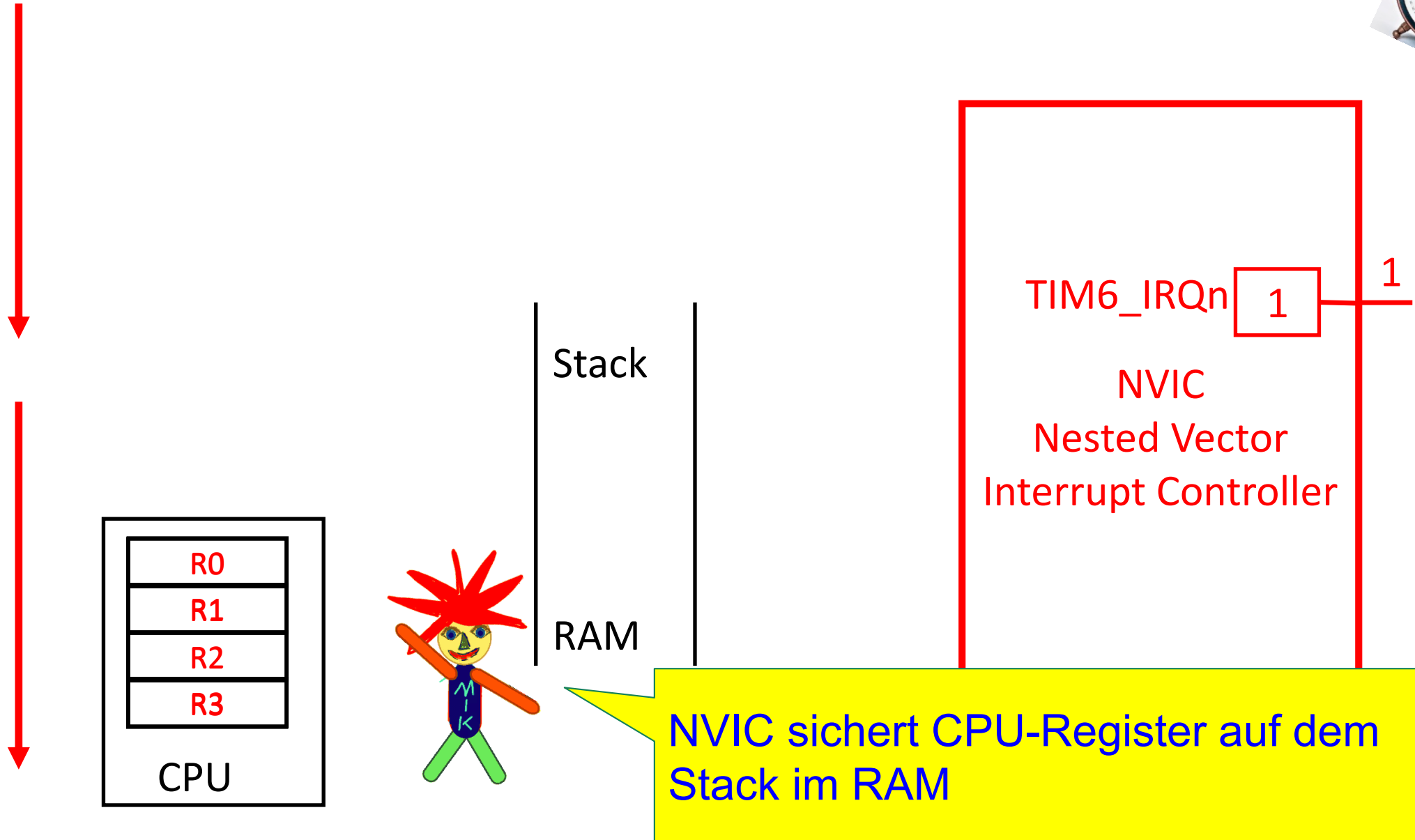
Timer und Interrupt

Hauptprogramm, Unterprogramm oder ISR mit niedrigerer Priorität



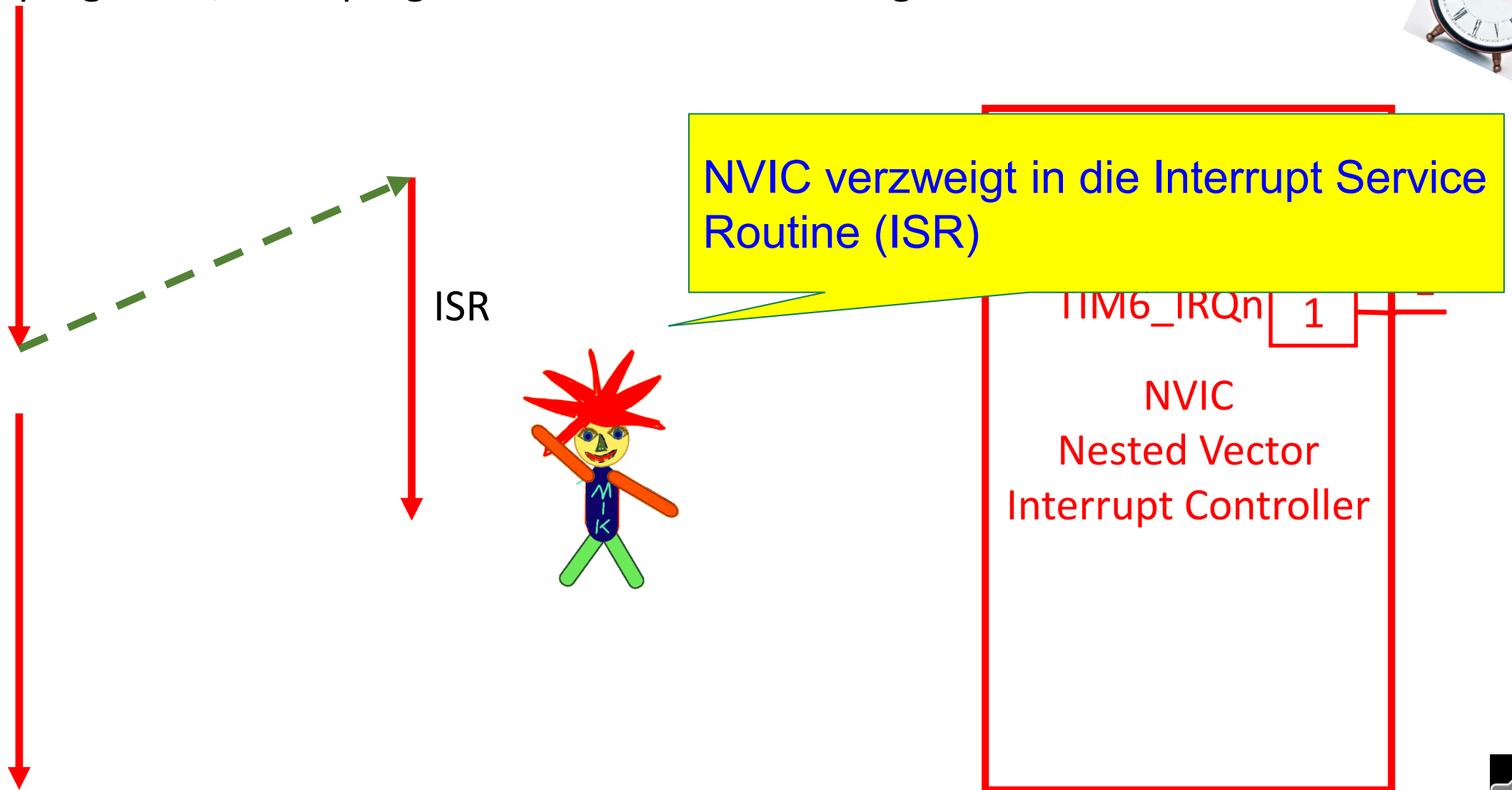
Timer und Interrupt

Hauptprogramm, Unterprogramm oder ISR mit niedrigerer Priorität



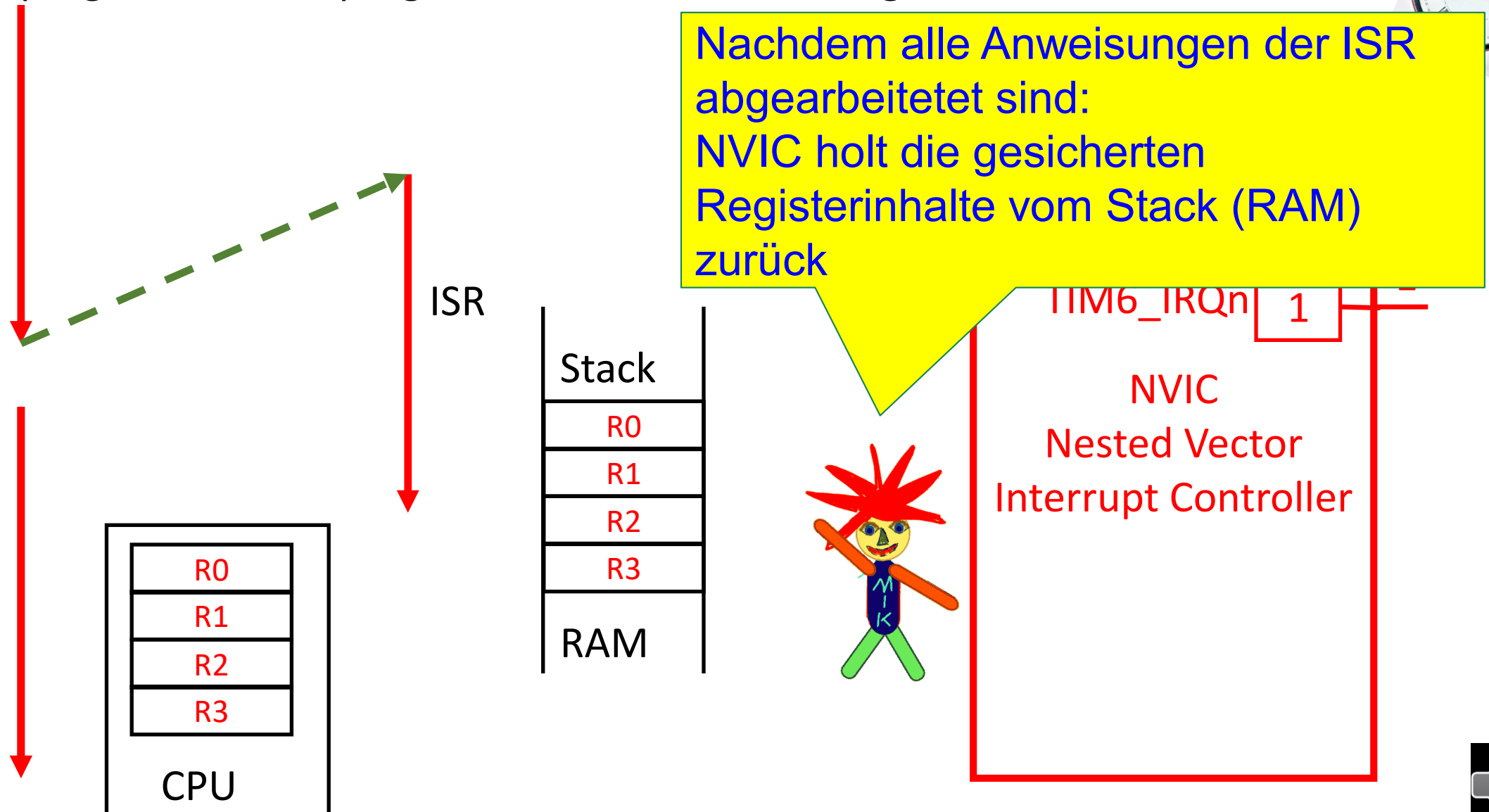
Timer und Interrupt

Hauptprogramm, Unterprogramm oder ISR mit niedrigerer Priorität



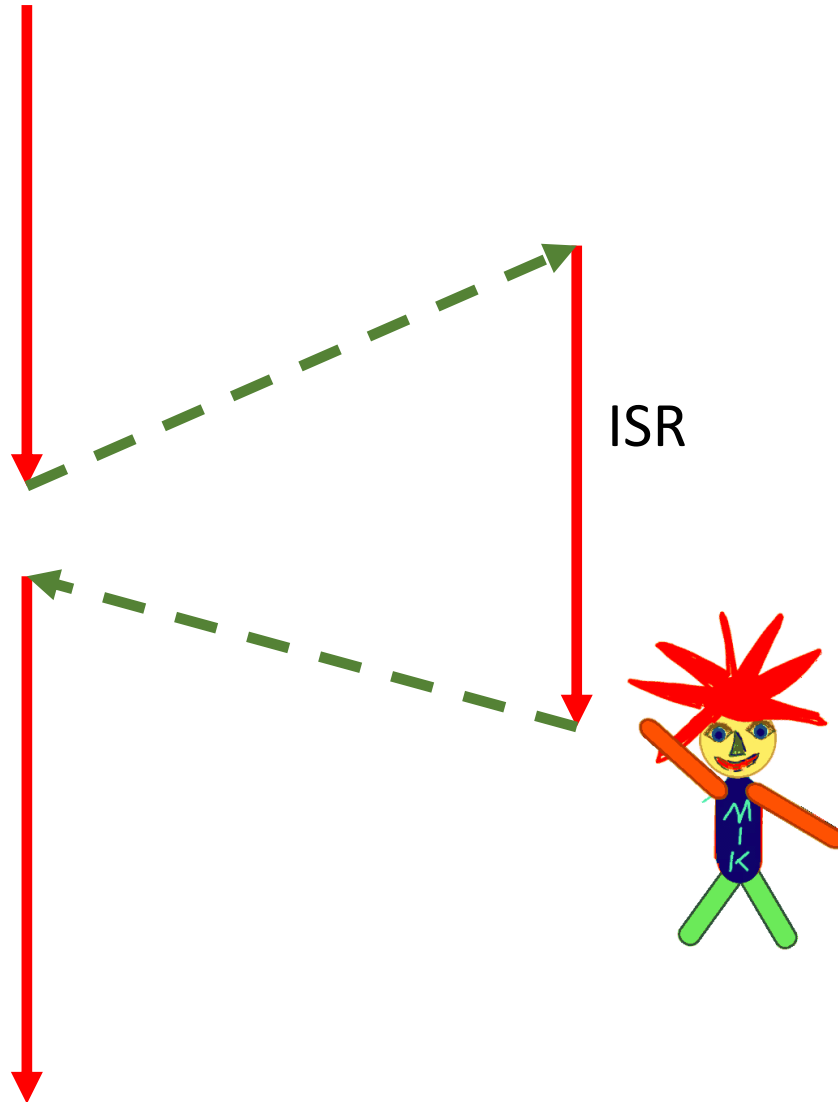
Timer und Interrupt

Hauptprogramm, Unterprogramm oder ISR mit niedrigerer Priorität



Timer und Interrupt

Hauptprogramm, Unterprogramm oder ISR mit niedrigerer Priorität

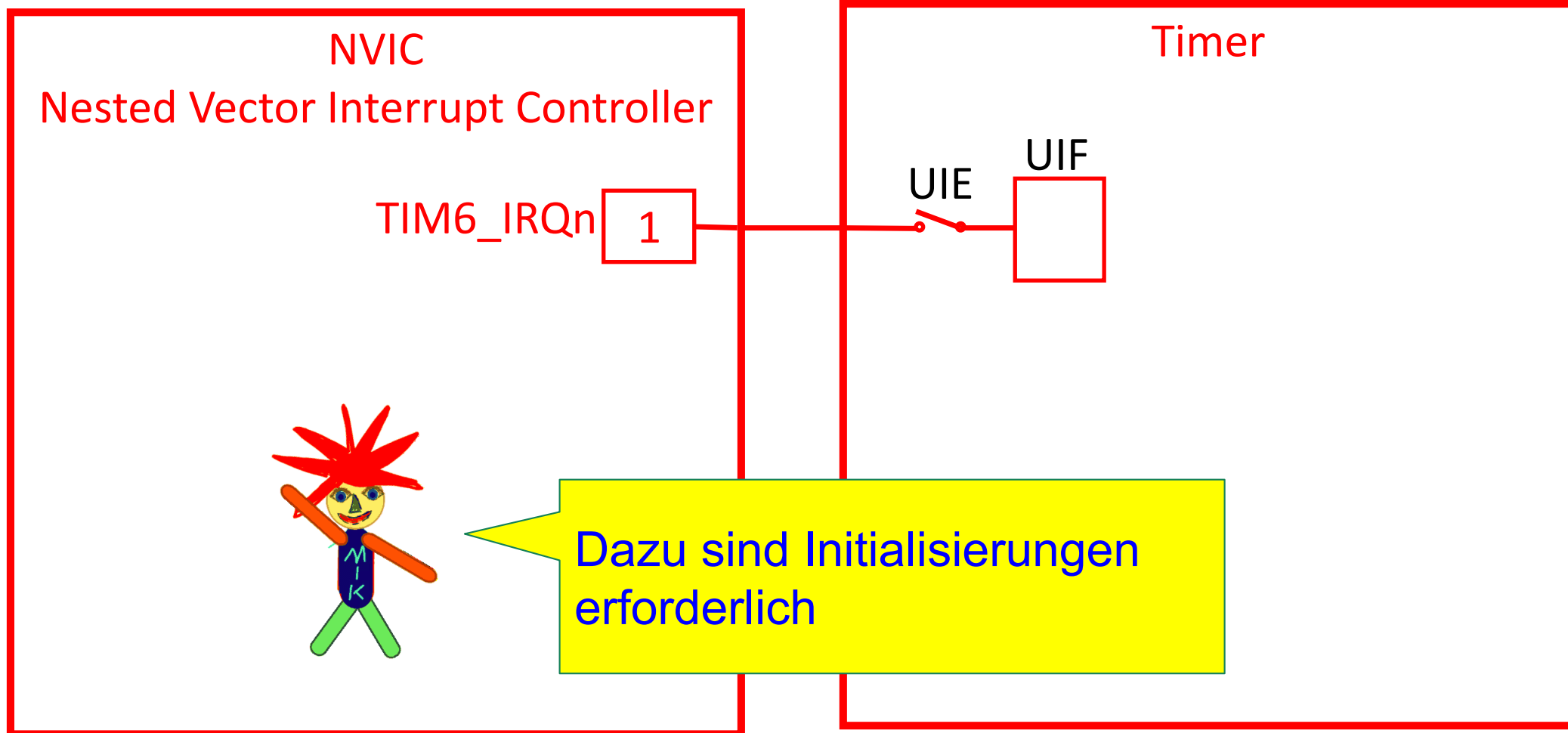


NVIC verzweigt zurück zum laufenden Programm

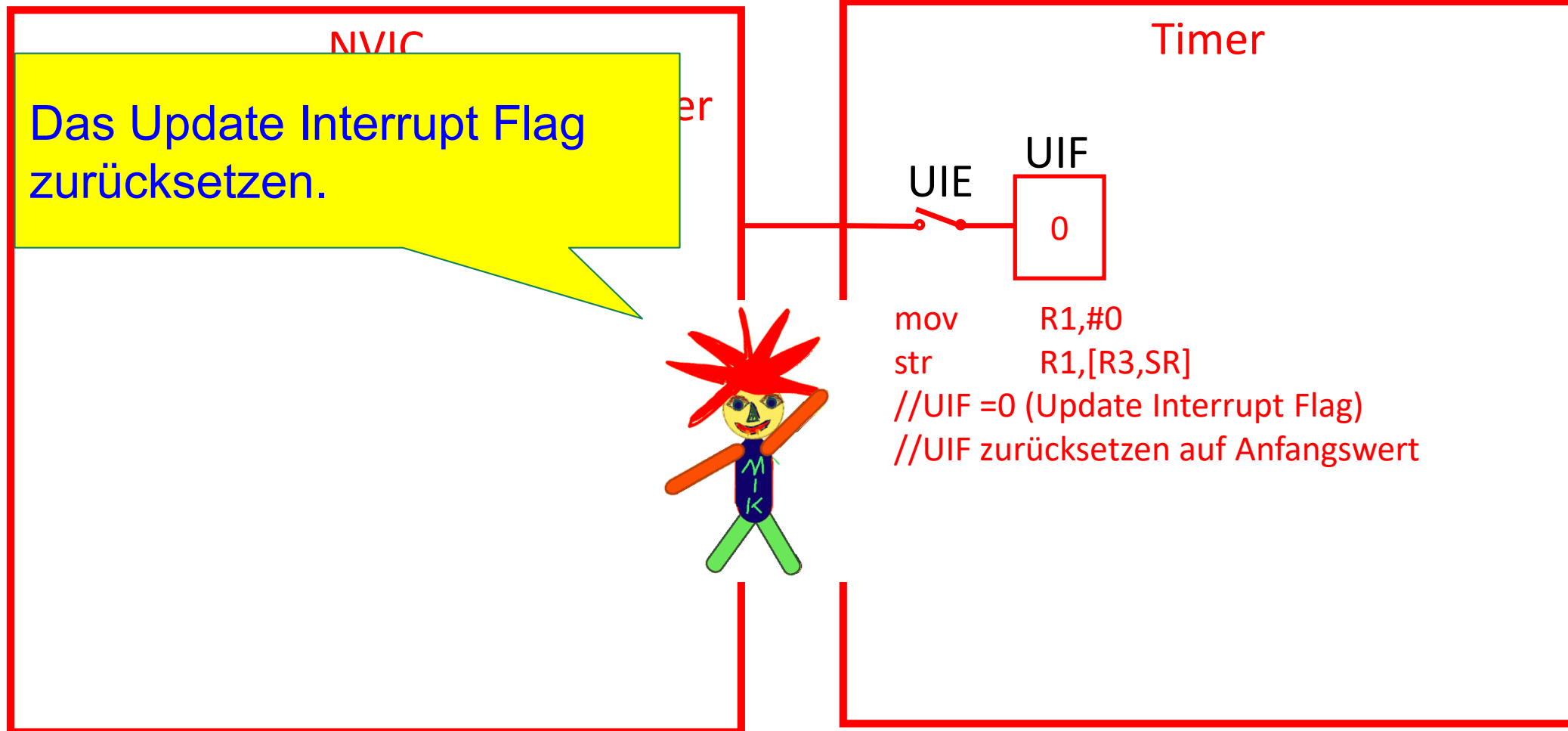
NVIC
Nested Vector
Interrupt Controller



Timer und Interrupt



Timer und Interrupt



Timer und Interrupt

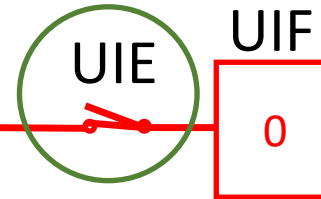


NVIC

Nested Vector Interrupt Controller

Timerinterrupt freigeben

Timer



//1. UIF =0 (Update Interrupt Flag)
//UIF zurücksetzen auf Anfangswert

```
mov    R1,#0
```

```
str    R1,[R3,SR]
```

//2. Timerinterrupt freigeben

```
movR1,#1
```

```
str R1,[R3,DIER]
```



Timer und Interrupt



NVIC

Nested Vector Interrupt Controller

```
.word RTC_Alarm_IRQHandler  
.word USB_FS_WKUP_IRQHandler  
.word isrTIM6+1//TIM6_IRQHandler  
.word isrTIM7+1//TIM7_IRQHandler  
.word 0  
.word TIM5_IRQHandler
```

startup_stm32l152retx.s



Die Anfangsadresse der ISR
in der Vektortabelle
eintragen, an der Stelle
(TIM6_IRQn) die für z.B.
TIM6 reserviert ist

Timer TIM6

UIE

UIF

0

//1. UIF =0 (Update Interrupt Flag)
//UIF zurücksetzen auf Anfangswert

Vektor-
tabelle

isrTIM6+1

TIM6_IRQn

isrTIM7+1

TIM7_IRQn

RAM



Timer und Interrupt



NVIC

Nested Vector Interrupt Controller

```
.word RTC_Alarm_IRQHandler  
.word USB_FS_WKUP_IRQHandler  
.word isrTIM6+1//TIM6_IRQHandler  
.word isrTIM7+1//TIM7_IRQHandler  
.word 0  
.word TIM5_IRQHandler
```



Über die Tabelle weiß NVIC,
welche ISR bei diesem
speziellen Interrupt gestartet
werden soll.

Timer TIM6

UIE

UIF

0

//1. UIF =0 (Update Interrupt Flag)
//UIF zurücksetzen auf Anfangswert

Vektor-
tabelle

isrTIM6+1

isrTIM7+1

RAM

TIM6_IRQn

TIM7_IRQn



Timer und Interrupt



NVIC

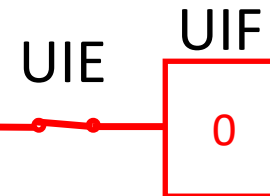
Nested Vector Interrupt Controller

```
ldr r4,=nvic  
ldr r1,=0xFFFFFFFF  
str r1,[r4,isrEnableReg0]  
str r1,[r4,isrEnableReg1]
```



Interruptfreigabe im NVIC

Timer TIM6



```
//1. UIF =0 (Update Interrupt Flag)  
//UIF zurücksetzen auf Anfangswert  
mov R1,#0  
str R1,[R3,SR]  
//2. Timerinterrupt freigeben  
movR1,#1  
str R1,[R3,DIER]
```



Timer und Interrupt



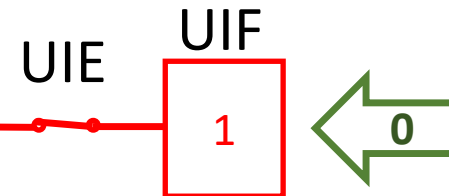
NVIC

Nested Vector Interrupt Controller



In der Interrupt Service
Routine (ISR)

Timer TIM6



//4. ISR schreiben (Timerauswahl in R3)

.global isrTIM6

.global isrTIM7

.global isrTIM2

isrTIM2:

isrTIM6:

isrTIM7:

mov R2,#0 //5. UIF zurücksetzen

str R2,[R3,SR] ...



Timer und Interrupt

