STM32F072 Nucleo-64, TIM16 problem

Goal: Generate a one-shot high pulse at port pin “P_LOAD” (PORTB.6, Nucleo CN5.3).
Nucleo is running with 8 MHz coming from ST-Link part of board.

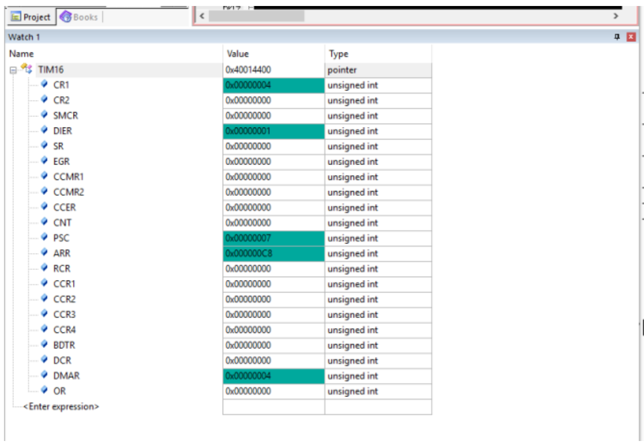
Port pin “P_LOAD” is initialized to “0 (we need high pulse).

```
    GPIOSetValue (P_LOAD, 0)
```

Set up of TIM16 follows.

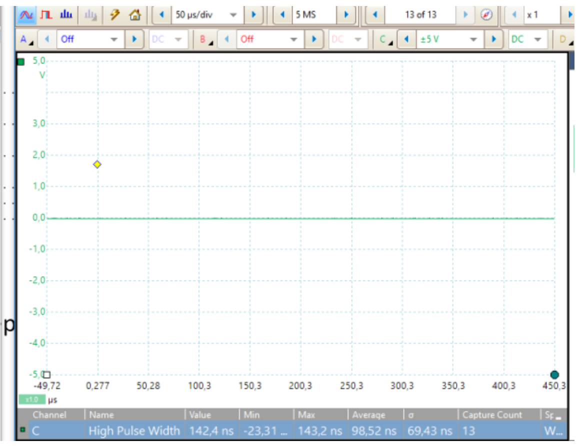
```
void Timer16_Init(uint32_t usec)
{

    //---- Disable and clear TIM16 IRQs (if any) in NVIC ----//
    //
    NVIC_DisableIRQ (TIM16_IRQn);
    NVIC_ClearPendingIRQ (TIM16_IRQn);


    //---- Enable APB clock for TIM16 (otherwise cannot write registers) ----//
    //
    RCC->APB2ENR |= RCC_APB2ENR_TIM16EN;


    //---- Set all registers to reset values ----//
    //
    TIM16->CR1    = 0;                                     // CEN := 0 => stop/disable timer
    TIM16->CR2    = 0;
    TIM16->DIER   = 0;
    TIM16->SR     = 0;
    TIM16->EGR    = 0;
    TIM16->CCMR1  = 0;
    TIM16->CCER   = 0;
    TIM16->CNT    = 0;
    TIM16->PSC    = 0;
    TIM16->ARR    = 0;
    TIM16->RCR    = 0;
    TIM16->CCR1   = 0;
    TIM16->BDTR   = 0;
    TIM16->DCR    = 0;
    TIM16->DMAR   = 0;
    TIM16->OR     = 0;


    //---- Set up for a 'usec' one shot timer ----//
    //
    if (SystemCoreClock == 8000000)
        TIM16->PSC =  7;                                   // Prescale value, dividing  8 MHz by  7+1= 8 => 1 MHz
    else
        TIM16->PSC = 47;                                   // Prescale value, dividing 48 MHz by 47+1=48 => 1 MHz

    TIM16->ARR = usec;                                     // Calculate ARR match value for a 'usec' one shot

//    TIM16->CR1   |= TIM_CR1_OPM;                           // OPM := 1 => enable one pulse mode
    TIM16->CR1   |= TIM_CR1_URS;                           // Interrupt on overflow/underflow only
    TIM16->DIER |= TIM_DIER_UIE;                           // Enable update interrupts

}// of void Timer16_Init(uint32_t usec)
```

After this initialization TIM16 registers and port pin “P_LOAD” look like this:



Watch 1

Name	Value	Type
TIM16	0x40014400	pointer
CR1	0x00000004	unsigned int
CR2	0x00000000	unsigned int
SMCR	0x00000000	unsigned int
DIER	0x00000001	unsigned int
SR	0x00000000	unsigned int
EGR	0x00000000	unsigned int
CCMR1	0x00000000	unsigned int
CCMR2	0x00000000	unsigned int
CCER	0x00000000	unsigned int
CNT	0x00000000	unsigned int
PSC	0x00000007	unsigned int
ARR	0x000000C8	unsigned int
RCR	0x00000000	unsigned int
CCR1	0x00000000	unsigned int
CCR2	0x00000000	unsigned int
CCR3	0x00000000	unsigned int
CCR4	0x00000000	unsigned int
BDTR	0x00000000	unsigned int
DCR	0x00000000	unsigned int
DMAR	0x00000004	unsigned int
OR	0x00000000	unsigned int
<Enter expression>		



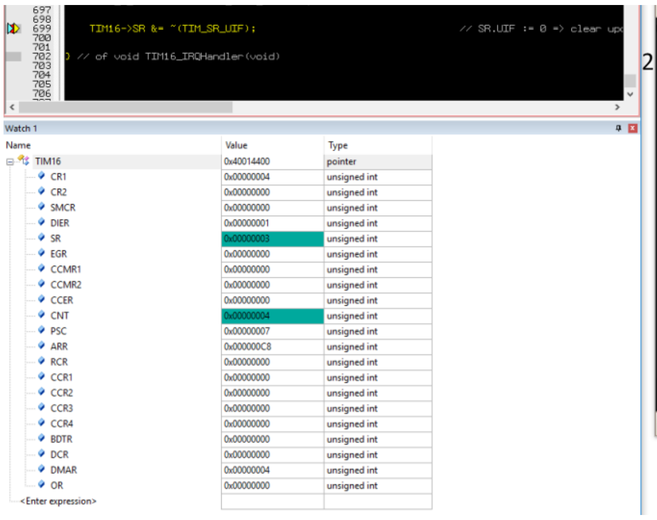
Channel	Name	Value	Min	Max	Average	σ	Capture Count	Sp…
C	High Pulse Width	142,4 ns	-23,31 …	143,2 ns	98,52 ns	69,43 ns	13	W…

To generate one shot high pulse at port pin “P_LOAD” (PORTB.6), I set up a timer start and a timer ISR routine like this.

```
void LOAD_Timer16_Start(uint32_t usec)
{
    if (usec > 16)
    {
        NVIC_DisableIRQ (TIM16_IRQn);                                   // Disable TIM16 interrupts

        GPIOSetValue(P_LOAD, 1);                                        // Generate rising edge

        TIM16->ARR  = usec;

        NVIC_EnableIRQ (TIM16_IRQn);                                    // ReEnable TIM16 interrupts

        TIM16->CNT  = 0;
        TIM16->CR1 |= TIM_CR1_CEN;                                      // CEN := 1 => enable timer
    }

} // of void LOAD_Timer16_Start(uint32_t usec)


void TIM16_IRQHandler(void)
{
    NVIC_DisableIRQ (TIM16_IRQn);                                   // Disable TIM16 interrupts (we are one shot....)
    TIM16->CR1 &= ~(TIM_CR1_CEN);                                   // CR1.CEN := 0 => stop/disable timer

    GPIOSetValue(P_LOAD, 0);                                        // Deactivate P_LOAD output again (high active)

    TIM16->SR &= ~(TIM_SR_UIF);                                     // SR.UIF := 0 => clear update interrupt flag bit

} // of void TIM16_IRQHandler(void)
```

Running this I get this pulse at “P_LOAD”

Stopping at end of ISR. There is a pulse, but not 200 µsec, only 42 µsec…

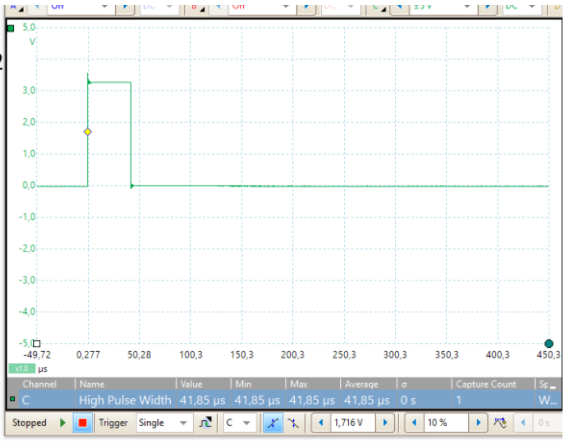


```
    TIM16->SR &= ~(TIM_SR_UIF);                                     // SR.UIF := 0 => clear upd

} // of void TIM16_IRQHandler(void)
```

Watch 1

Name	Value	Type
TIM16	0x40014400	pointer
CR1	0x00000004	unsigned int
CR2	0x00000000	unsigned int
SMCR	0x00000000	unsigned int
DIER	0x00000001	unsigned int
SR	0x00000003	unsigned int
EGR	0x00000000	unsigned int
CCMR1	0x00000000	unsigned int
CCMR2	0x00000000	unsigned int
CCER	0x00000000	unsigned int
CNT	0x00000004	unsigned int
PSC	0x00000007	unsigned int
ARR	0x000000C8	unsigned int
RCR	0x00000000	unsigned int
CCR1	0x00000000	unsigned int
CCR2	0x00000000	unsigned int
CCR3	0x00000000	unsigned int
CCR4	0x00000000	unsigned int
BDTR	0x00000000	unsigned int
DCR	0x00000000	unsigned int
DMAR	0x00000004	unsigned int
OR	0x00000000	unsigned int
<Enter expression>		

2



Channel	Name	Value	Min	Max	Average	σ	Capture Count	Sp…
C	High Pulse Width	41,85 µs	41,85 µs	41,85 µs	41,85 µs	0 s	1	W…

Checking TIM16 registervalues.
SR shows 0x03

0x01	SR.UIF == 1	=> Update interrupt pending

(which is what we expect)

0x02	SR.CC1F == 1	=> The content of the counter TIMx_CNT matches the TIMx_CCR1 register

(CC1F was initialized to 0, but this flag should do no harm, because we had enable UIF interrupts only)

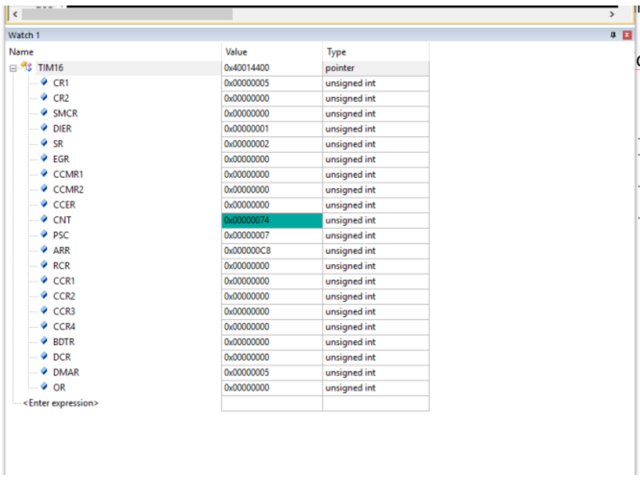
CNT shows 0x04 => latency, CNT runs until counter stopped in ISR

To get a periodical timer, I modified the TIM16_IRQHandler:

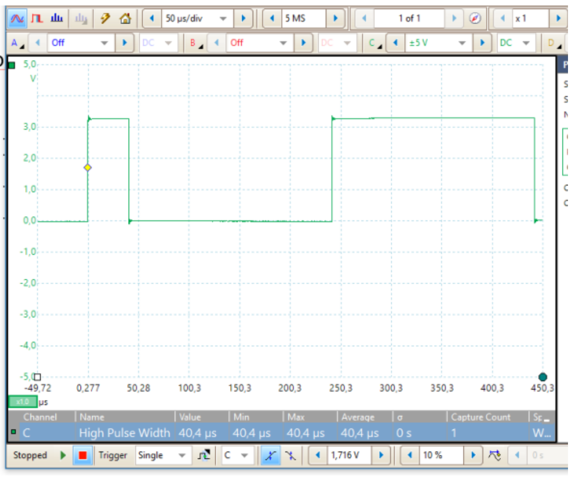
- no longer disable further TIM16 interrupts,
- no longer stop timer,
- change GPIO from GPIOSetValue() to GPIOToggleValue()

```
void TIM16_IRQHandler(void)
{
//    NVIC_DisableIRQ (TIM16_IRQn);                          // Disable TIM16 interrupts (we are one shot....)
//    TIM16->CR1 &= ~(TIM_CR1_CEN);                          // CR1.CEN := 0 => stop/disable timer

    GPIOToggleValue(P_LOAD, 0);                            // TEST: Toggle P_LOAD output

    TIM16->SR &= ~(TIM_SR_UIF);                            // SR.UIF := 0 => clear update interrupt flag bit

} // of void TIM16_IRQHandler(void)
```

Now I get this

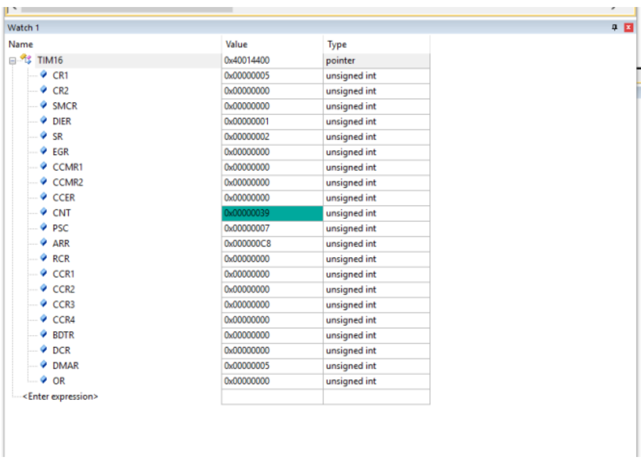

Watch 1

Name	Value	Type
TIM16	0x40014400	pointer
CR1	0x00000005	unsigned int
CR2	0x00000000	unsigned int
SMCR	0x00000000	unsigned int
DIER	0x00000001	unsigned int
SR	0x00000002	unsigned int
EGR	0x00000000	unsigned int
CCMR1	0x00000000	unsigned int
CCMR2	0x00000000	unsigned int
CCER	0x00000000	unsigned int
CNT	0x00000074	unsigned int
PSC	0x00000007	unsigned int
ARR	0x000000C8	unsigned int
RCR	0x00000000	unsigned int
CCR1	0x00000000	unsigned int
CCR2	0x00000000	unsigned int
CCR3	0x00000000	unsigned int
CCR4	0x00000000	unsigned int
BDTR	0x00000000	unsigned int
DCR	0x00000000	unsigned int
DMAR	0x00000005	unsigned int
OR	0x00000000	unsigned int
<Enter expression>		

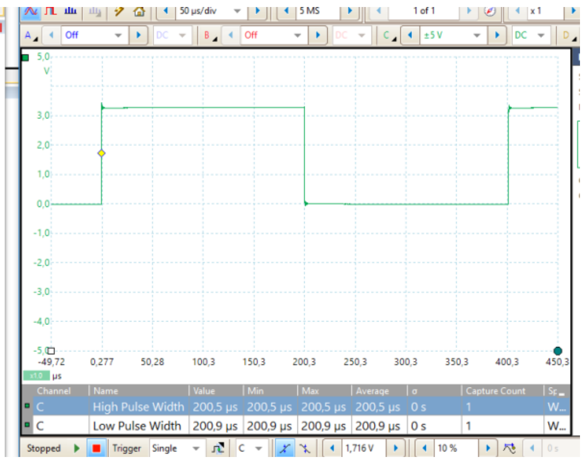


Channel	Name	Value	Min	Max	Average	σ	Capture Count	Sp...
C	High Pulse Width	40,4 µs	40,4 µs	40,4 µs	40,4 µs	0 s	1	W...

Second and all further high and low pulses are 200 µsec as expected…


Watch 1

Name	Value	Type
TIM16	0x40014400	pointer
CR1	0x00000005	unsigned int
CR2	0x00000000	unsigned int
SMCR	0x00000000	unsigned int
DIER	0x00000001	unsigned int
SR	0x00000002	unsigned int
EGR	0x00000000	unsigned int
CCMR1	0x00000000	unsigned int
CCMR2	0x00000000	unsigned int
CCER	0x00000000	unsigned int
CNT	0x00000039	unsigned int
PSC	0x00000007	unsigned int
ARR	0x000000C8	unsigned int
RCR	0x00000000	unsigned int
CCR1	0x00000000	unsigned int
CCR2	0x00000000	unsigned int
CCR3	0x00000000	unsigned int
CCR4	0x00000000	unsigned int
BDTR	0x00000000	unsigned int
DCR	0x00000000	unsigned int
DMAR	0x00000005	unsigned int
OR	0x00000000	unsigned int
<Enter expression>		



Channel	Name	Value	Min	Max	Average	σ	Capture Count	Sp...
C	High Pulse Width	200,5 µs	200,5 µs	200,5 µs	200,5 µs	0 s	1	W...
C	Low Pulse Width	200,9 µs	200,9 µs	200,9 µs	200,9 µs	0 s	1	W...