17. Advanced-control timer (TIM1)

17.1. TIM1 introduction

The advanced-control timers (TIM1) consist of a 16-bit auto-reload counter driven by a programmable prescaler. It may be used for a variety of purposes, including measuring the pulse lengths of input signals (input capture) or generating output waveforms (output compare, PWM, complementary PWM with dead-time insertion).

Pulse lengths and waveform periods can be modulated from a few microseconds to several milliseconds using the timer prescaler and the RCC clock controller prescalers. The advanced-control (TIM1) and general-purpose (TIMx) timers are completely independent, and do not share any resources. They can be synchronized together.

17.2. TIM1 main features

- 16-bit up, down, up/down auto-reload counter.
- 16-bit programmable prescaler allowing dividing (also “on the fly”) the counter clock frequency either by any factor between 1 and 65536.
- Up to 4 independent channels for:
  - Input capture
  - Output compare
  - PWM generation (Edge and Center-aligned Mode)
  - One-pulse mode output
- Complementary outputs with programmable dead-time
- Synchronization circuit to control the timer with external signals and to interconnect several timers together.
- Repetition counter to update the timer registers only after a given number of cycles of the counter.
- Break input to put the timer’s output signals in reset state or in a known state.
- Interrupt/DMA generation on the following events:
  - Update: counter overflow/underflow, counter initialization (by software or internal/external trigger)
  - Trigger event (counter start, stop, initialization or count by internal/external trigger)
  - Input capture
  - Output compare
  - Break input
- Supports incremental (quadrature) encoder and hall-sensor circuitry for positioning purposes
- Trigger input for external clock or cycle-by-cycle current management

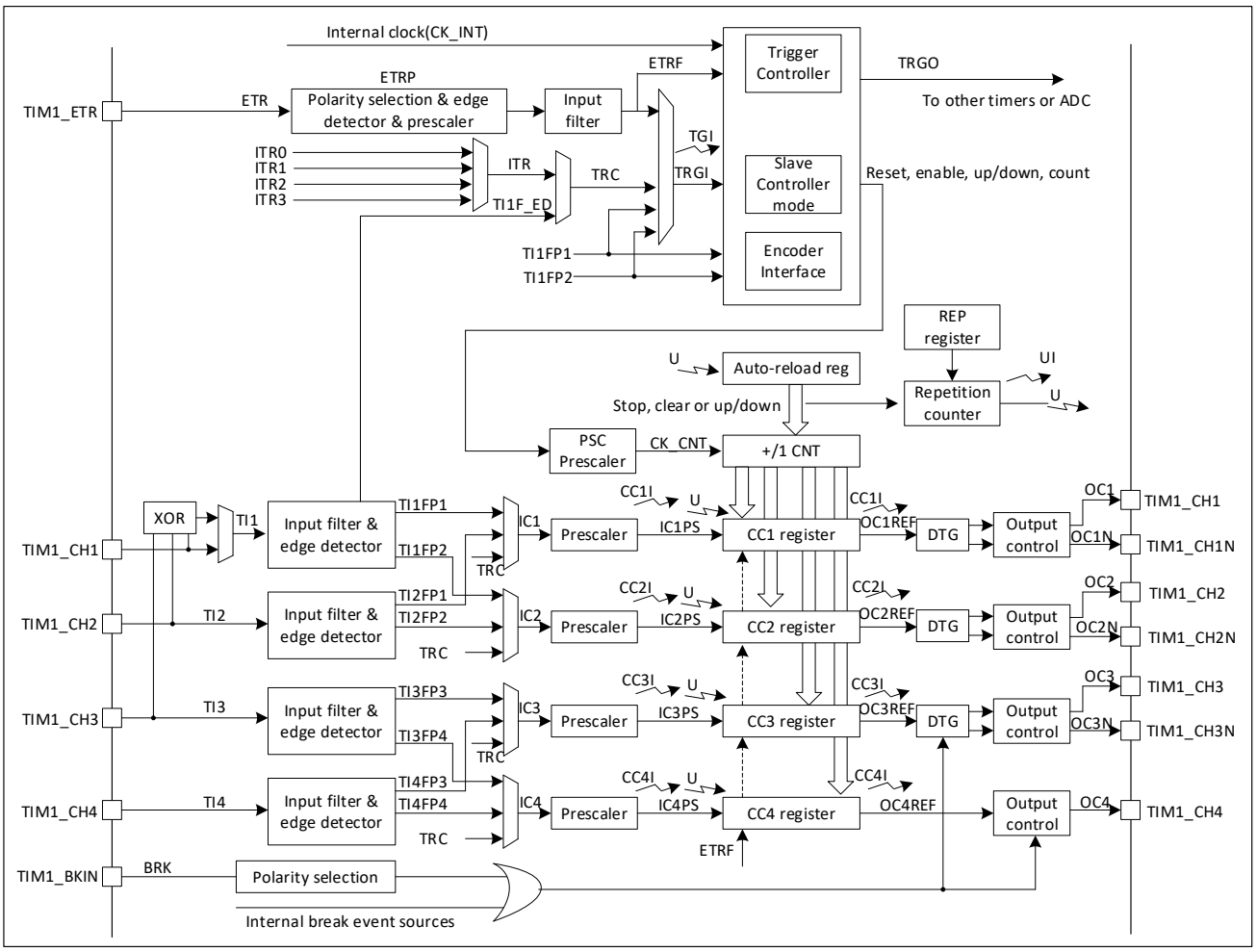

Figure 17-1 Advanced-control timer block diagram

Notes:

Reg — Preload registers transferred to active registers on U event according to control bit

Event

Interrupt & DMA output

17.3. TIM1 functional description

17.3.1. Time-base unit

The main block of the programmable advanced-control timer is a 16-bit counter with its related auto-reload register. The counter can count up, down or both up and down. The counter clock can be divided by a prescaler. The counter, the auto-reload register and the prescaler register can be written or read by software. This is true even when the counter is running.

The time-base unit includes:

- Counter register (TIMx_CNT)
- Prescaler register (TIMx_PSC)
- Auto-reload register (TIMx_ARR)
- Repetition counter register (TIMx_RCR)

The auto-reload register is preloaded. Writing to or reading from the auto-reload register accesses the preload register. The content of the preload register are transferred into the shadow register permanently or at each update event (UEV), depending on the auto-reload preload enable bit (ARPE) in TIMx_CR1 register. The update event is sent when the counter reaches the overflow (or underflow when downcounting) and if the UDIS bit equals 0 in the TIMx_CR1 register. It can also be generated by software.

The counter is clocked by the prescaler output CK_CNT, which is enabled only when the counter enable bit (CEN) in TIMx_CR1 register is set.

Note that the counter starts counting 1 clock cycle after setting the CEN bit in the TIMx_CR1 register.

Prescaler description

The prescaler can divide the counter clock frequency by any factor between 1 and 65536. It is based on a 16-bit counter controlled through a 16-bit register (in the TIMx_PSC register).

It can be changed on the fly as this control register is buffered. The new prescaler ratio is taken into account at the next update event.

Figure 17-2 and Figure 17-3 give some examples of the counter behavior when the prescaler ratio is changed on the fly:

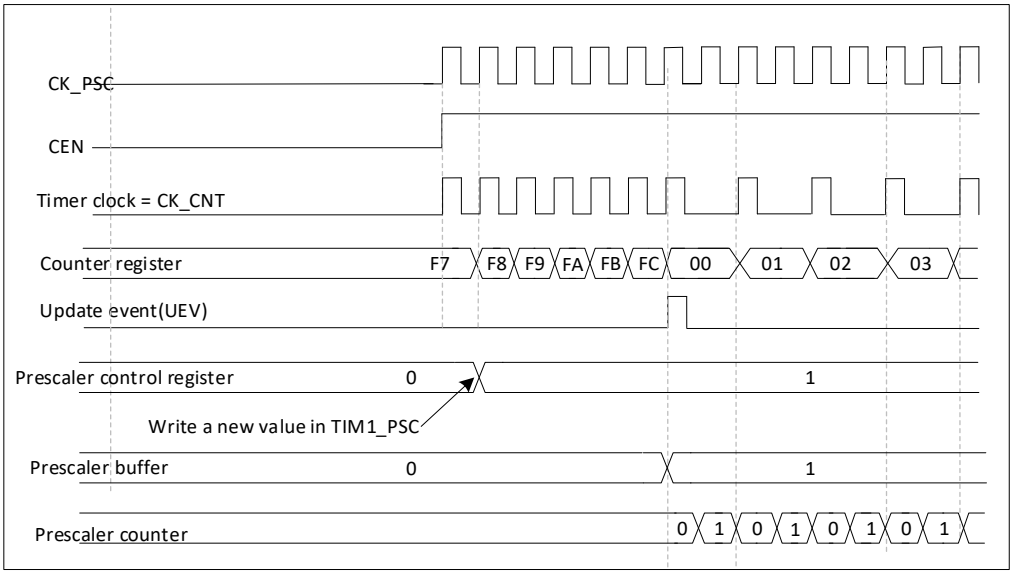

Figure 17-2 Counter timing diagram with prescaler division change from 1 to 2

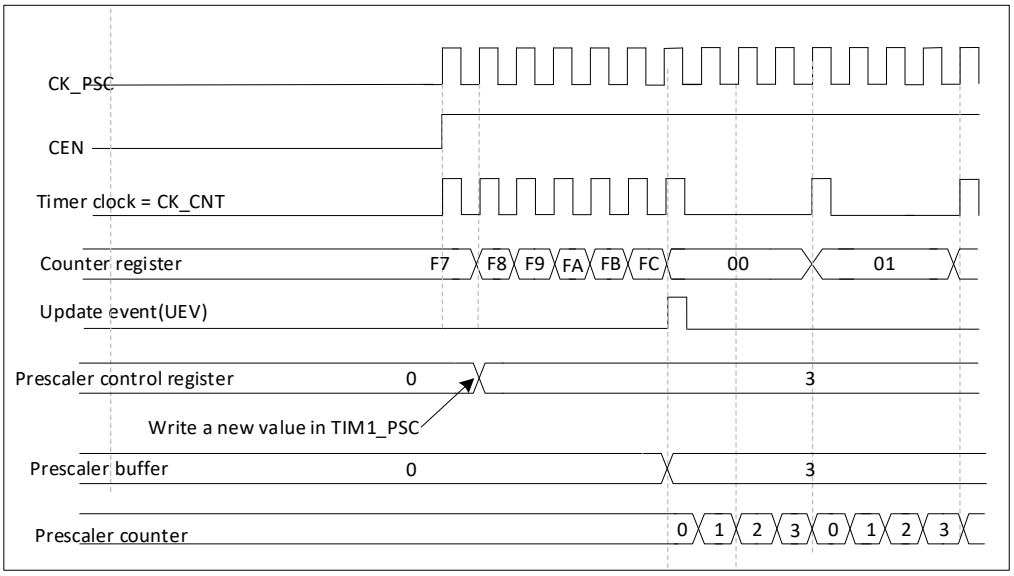

Figure 17-3 Counter timing diagram with prescaler division change from 1 to 4

17.3.2. Counter modes

Upcounting mode

In upcounting mode, the counter counts from 0 to the auto-reload value (content of the TIMx_ARR register), then restarts from 0 and generates a counter overflow event.

If the repetition counter is used, the update event (UEV) is generated after upcounting is repeated for the number of times programmed in the repetition counter register plus one (TIMx_RCR+1). Else the update event is generated at each counter overflow.

Setting the UG bit in the TIMx_EGR register (by software or by using the slave mode controller) also generates an update event.

The UEV event can be disabled by software by setting the UDIS bit in the TIMx_CR1 register. This is to avoid updating the shadow registers while writing new values in the preload registers. Then no update event occurs until the UDIS bit has been written to 0.

However, the counter restarts from 0, as well as the counter of the prescaler (but the prescale rate does not change). In addition, if the URS bit (update request selection) in TIMx_CR1 register is set, setting the UG bit generates an update event UEV but without setting the UIF flag (thus no interrupt or DMA request is sent). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

When an update event occurs, all the registers are updated and the update flag (UIF bit in TIMx_SR register) is set (depending on the URS bit):

- The repetition counter is reloaded with the content of TIMx_RCR register
- The auto-reload shadow register is updated with the preload value (TIMx_ARR)
- The buffer of the prescaler is reloaded with the preload value (content of the TIMx_PSCregister)

The following figures show some examples of the counter behavior for different clock frequencies when TIMx_ARR = 0x36.

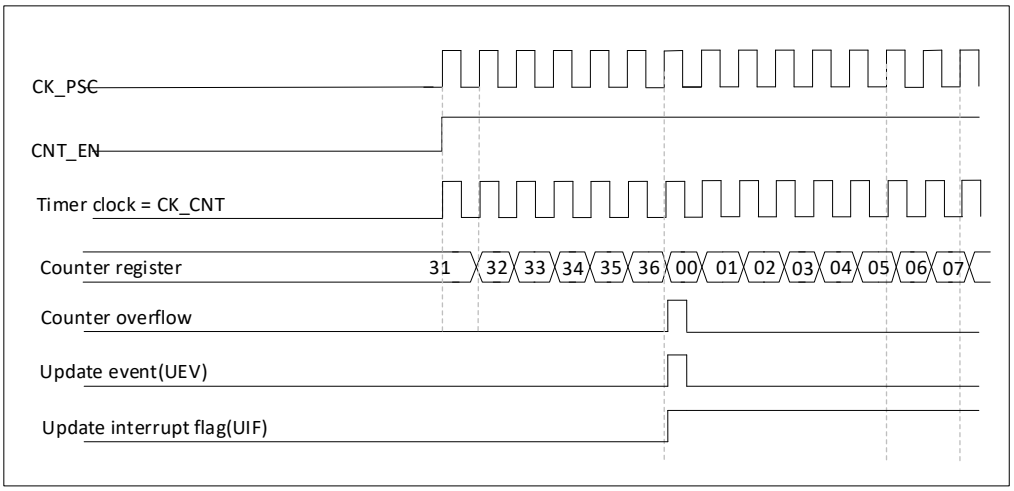

Figure 17-4 Counter timing diagram, internal clock divided by 1

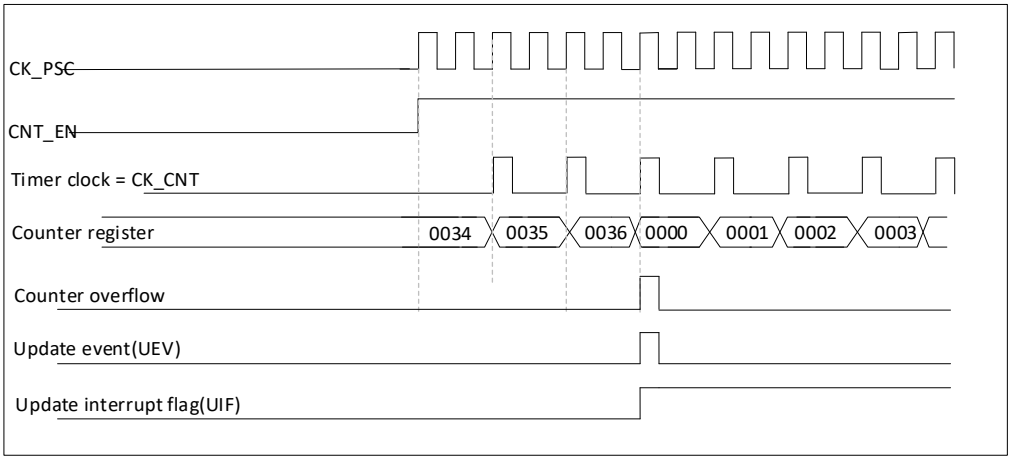

Figure 17-5 Counter timing diagram, internal clock divided by 2

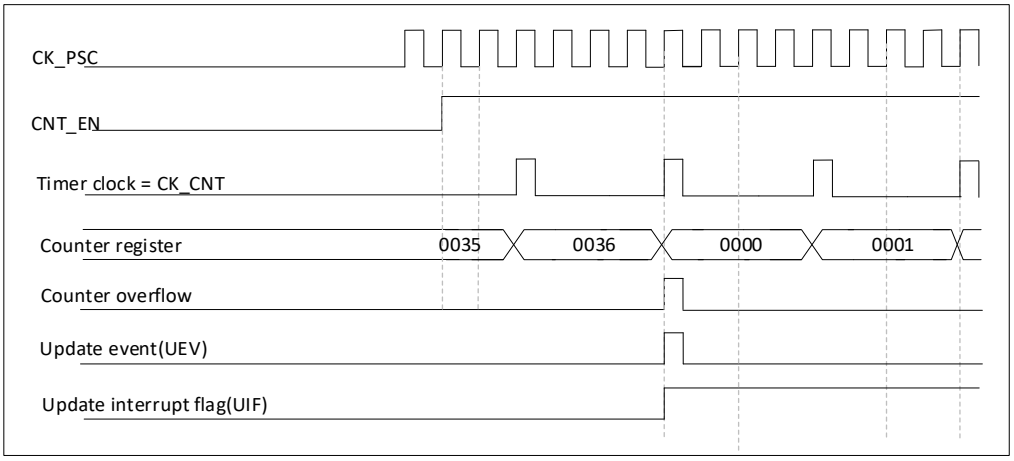

Figure 17-6 Counter timing diagram, internal clock divided by 4

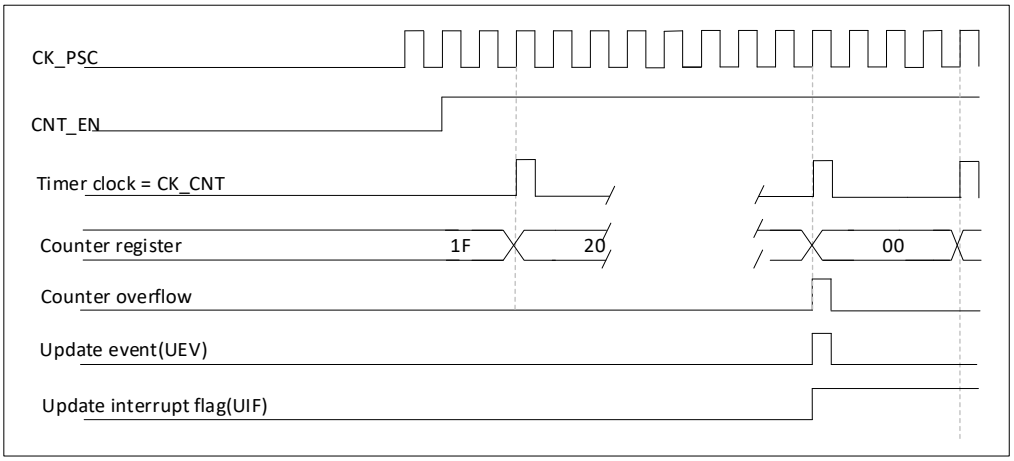

Figure 17-7 Counter timing diagram, internal clock divided by N

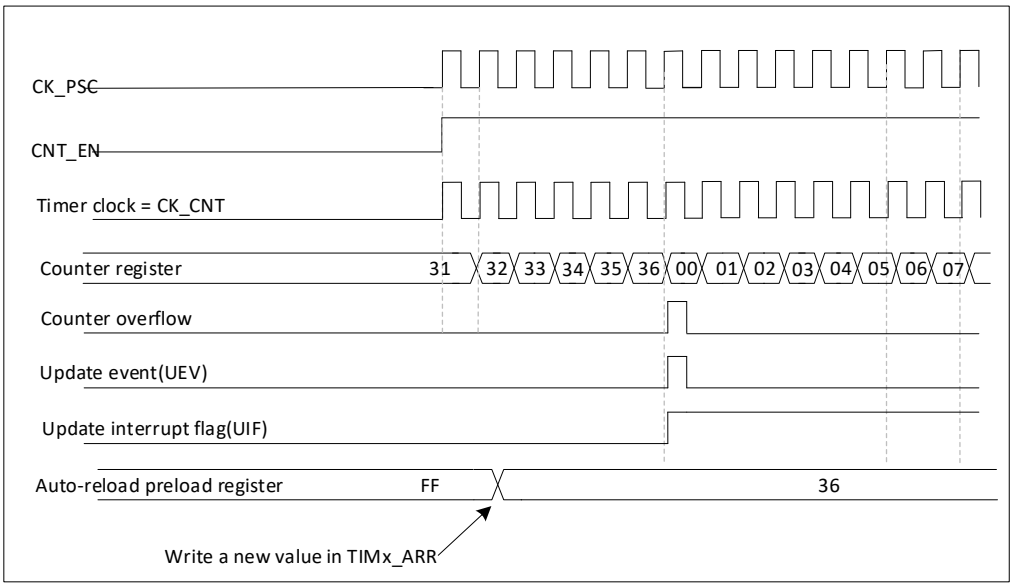

Figure 17-8 Counter timing diagram, update event when ARPE = 0 (TIMx_ARR no preloaded)

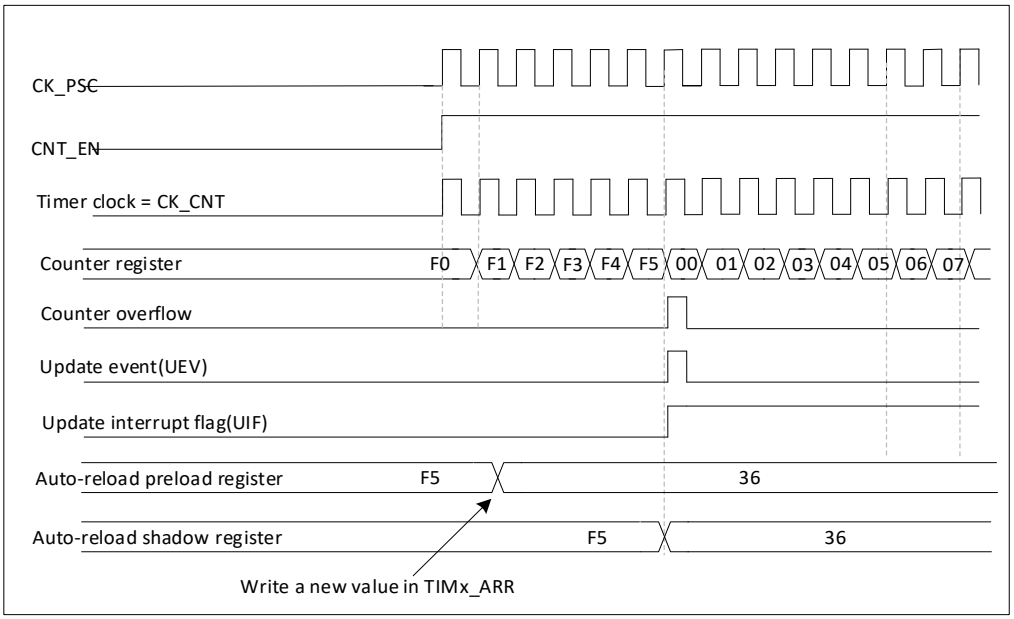

Figure 17-9 Counter timing diagram, update event when ARPE = 1 (TIMx_ARR preloaded)

Downcounting mode

In downcounting mode, the counter counts from the auto-reload value (content of the TIMx_ARR register) down to 0, then restarts from the auto-reload value and generates a counter underflow event.

If the repetition counter is used, the update event (UEV) is generated after downcounting is repeated for the number of times programmed in the repetition counter register (TIMx_RCR). Else the update event is generated at each counter underflow.

Setting the UG bit in the TIMx_EGR register (by software or by using the slave mode controller) also generates an update event.

The UEV update event can be disabled by software by setting the UDIS bit in TIMx_CR1 register. This is to avoid updating the shadow registers while writing new values in the preload registers. Then no update event occurs until UDIS bit has been written to 0.

However, the counter restarts from the current auto-reload value, whereas the counter of the prescaler restarts from 0 (but the prescale rate doesn't change).

In addition, if the URS bit (update request selection) in TIMx_CR1 register is set, setting the UG bit generates an update event UEV but without setting the UIF flag (thus no interrupt or DMA request is sent). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

When an update event occurs, all the registers are updated and the update flag (UIF bit in TIMx_SR register) is set (depending on the URS bit):

- The repetition counter is reloaded with the content of TIMx_RCR register
- The buffer of the prescaler is reloaded with the preload value (content of the TIMx_PSC register)
- The auto-reload active register is updated with the preload value (content of the TIMx_ARR register)

Note: the auto-reload is updated before the counter is reloaded, so that the next period is the expected one.

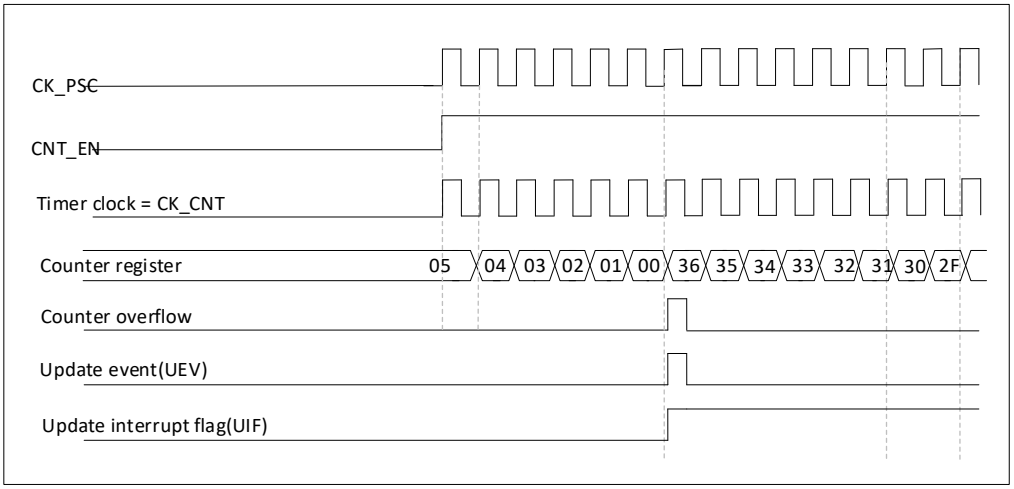

Figure 17-10 Counter timing diagram, internal clock divided by 1

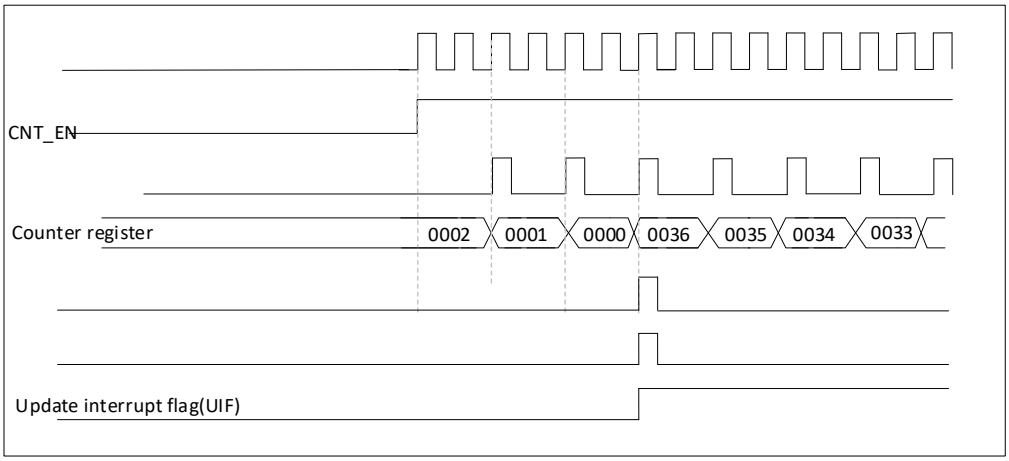

Figure 17-11 Counter timing diagram, internal clock divided by 2

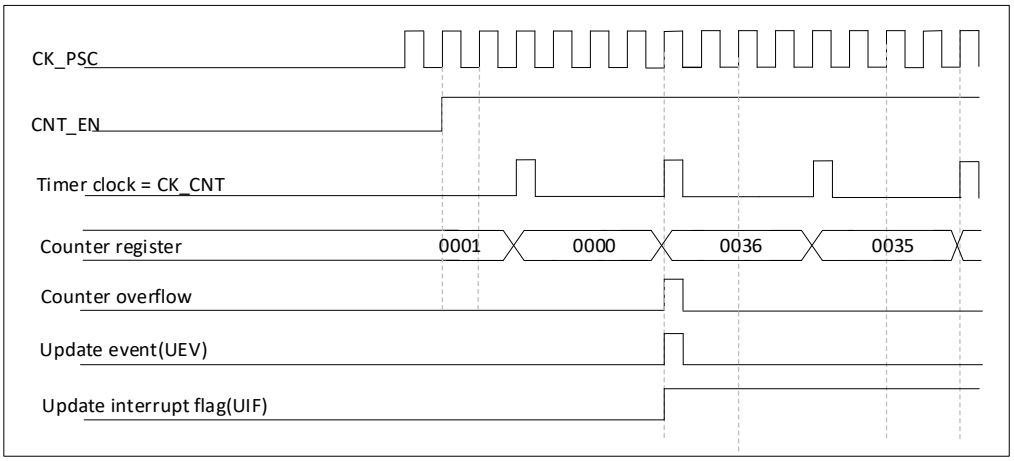

Figure 17-12 Counter timing diagram, internal clock divided by 4

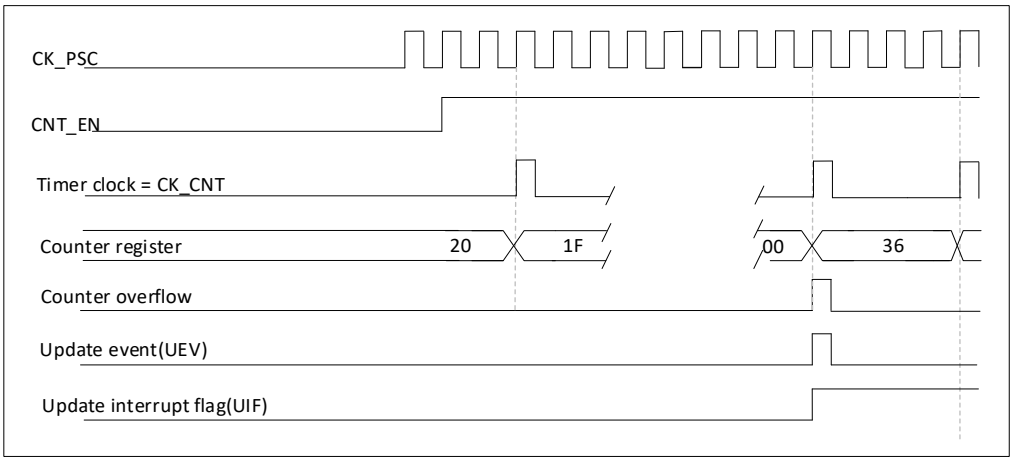

Figure 17-13 Counter timing diagram, internal clock divided by N

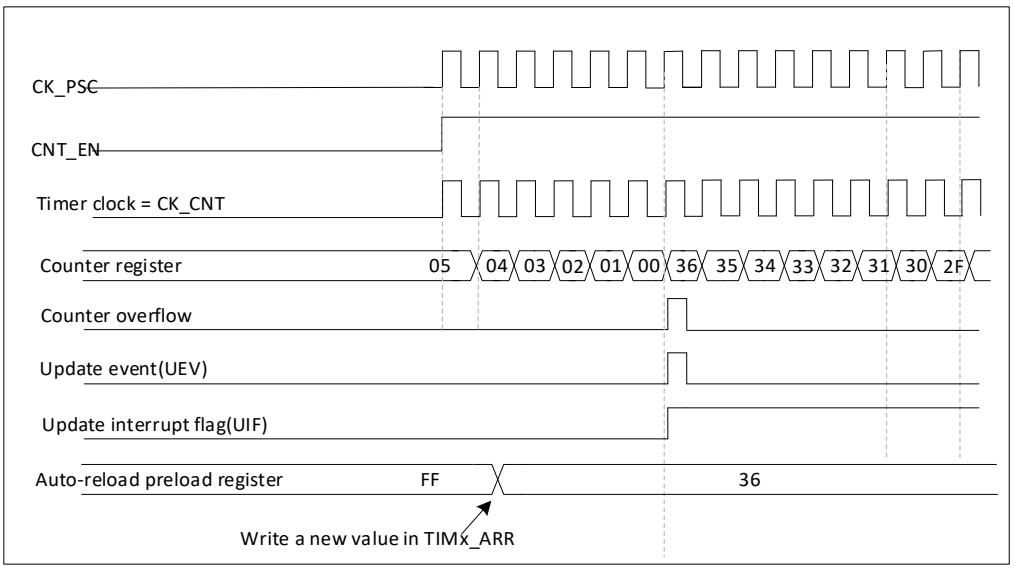

Figure 17-14 Counter timing diagram, update event when repetition counter is not used

Center-aligned mode (up/down counting)

In center-aligned mode, the counter counts from 0 to the auto-reload value (content of the TIMx_ARR register) – 1, generates a counter overflow event, then counts from the autoreload value down to 1 and generates a counter underflow event. Then it restarts counting from 0.

Center-aligned mode is active when the CMS bits in TIMx_CR1 register are not equal to '00'. The Output compare interrupt flag of channels configured in output is set when: the counter counts down (Center aligned mode 1, CMS = "01"), the counter counts up (Center aligned mode 2, CMS = "10") the counter counts up and down (Center aligned mode 3, CMS = "11").

In this mode, the DIR direction bit in the TIMx_CR1 register cannot be written. It is updated by hardware and gives the current direction of the counter.

The update event can be generated at each counter overflow and at each counter underflow or by setting the UG bit in the TIMx_EGR register (by software or by using the slave mode controller) also generates an update event. In this case, the counter restarts counting from 0, as well as the counter of the prescaler.

The UEV update event can be disabled by software by setting the UDIS bit in the TIMx_CR1 register. This is to avoid updating the shadow registers while writing new values in the preload registers. Then no update event occurs until UDIS bit has been written to 0.

However, the counter continues counting up and down, based on the current auto-reload value.

In addition, if the URS bit (update request selection) in TIMx_CR1 register is set, setting the UG bit generates an UEV update event but without setting the UIF flag (thus no interrupt or DMA request is sent). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

When an update event occurs, all the registers are updated and the update flag (UIF bit in TIMx_SR register) is set (depending on the URS bit).

- The repetition counter is reloaded with the content of TIMx_RCR register
- The buffer of the prescaler is reloaded with the preload value (content of the TIMx_PSC register)
- The auto-reload active register is updated with the preload value (content of the TIMx_ARR register)

Note: if the update source is a counter overflow, the autoreload is updated before the counter is reloaded, so that the next period is the expected one (the counter is loaded with the new value).

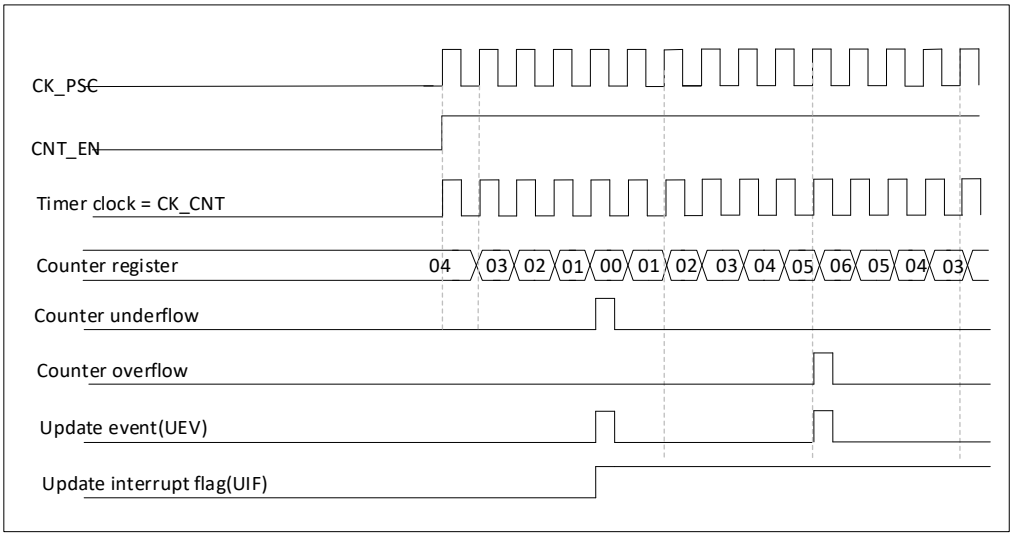

Figure 17-15 Counter timing diagram, internal clock divided by 1, TIMx_ARR = 0x6

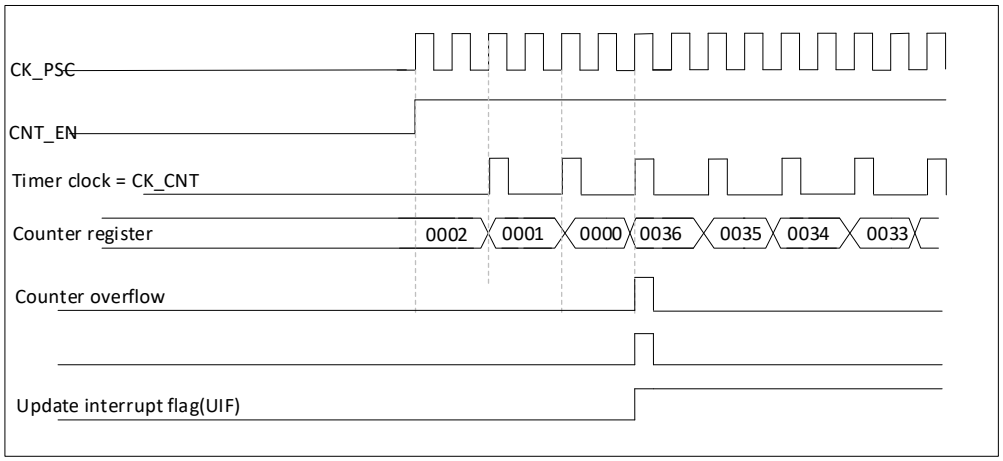

Figure 17-16 Counter timing diagram, internal clock divided by 2, TIMx_ARR = 0x36

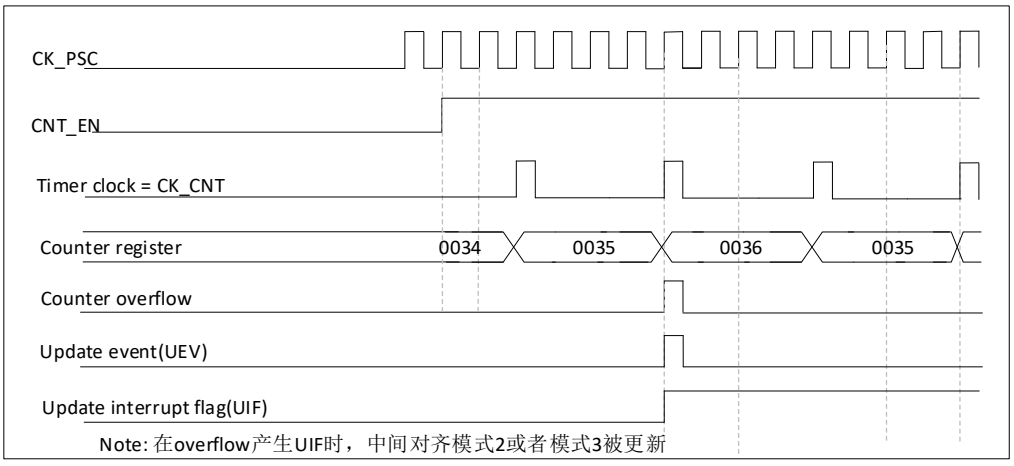

Figure 17-17 Counter timing diagram, internal clock divided by 4, TIMx_ARR = 0x36

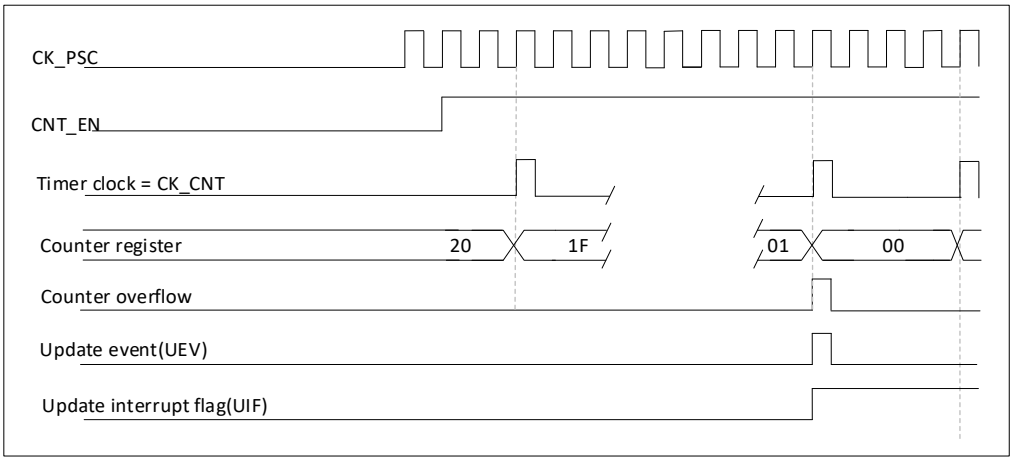

Figure 17-18 Counter timing diagram, internal clock divided by N

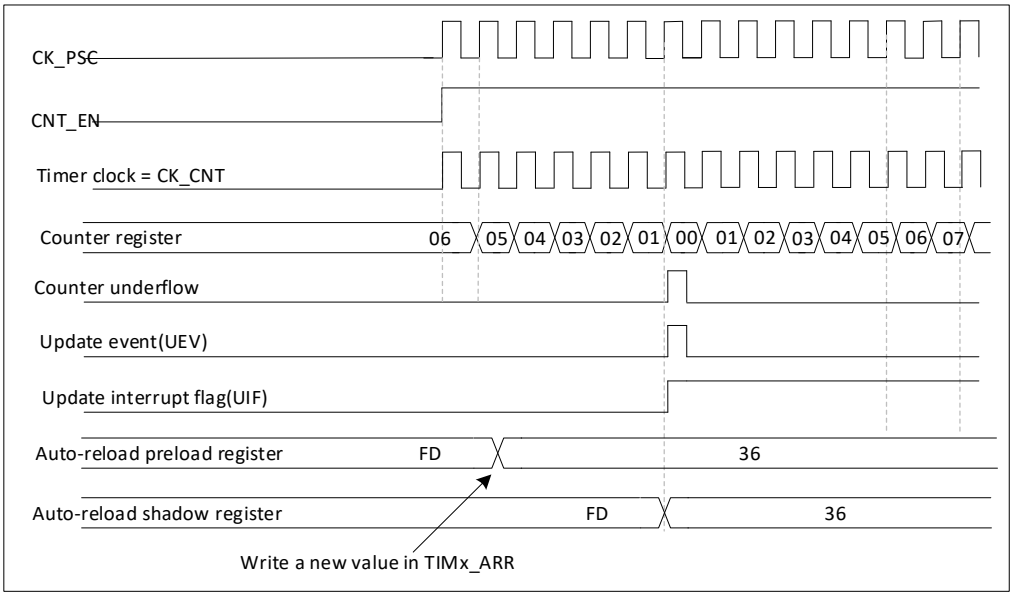

Figure 17-19 Counter timing diagram, update event with ARPE = 1 (counter underflow)

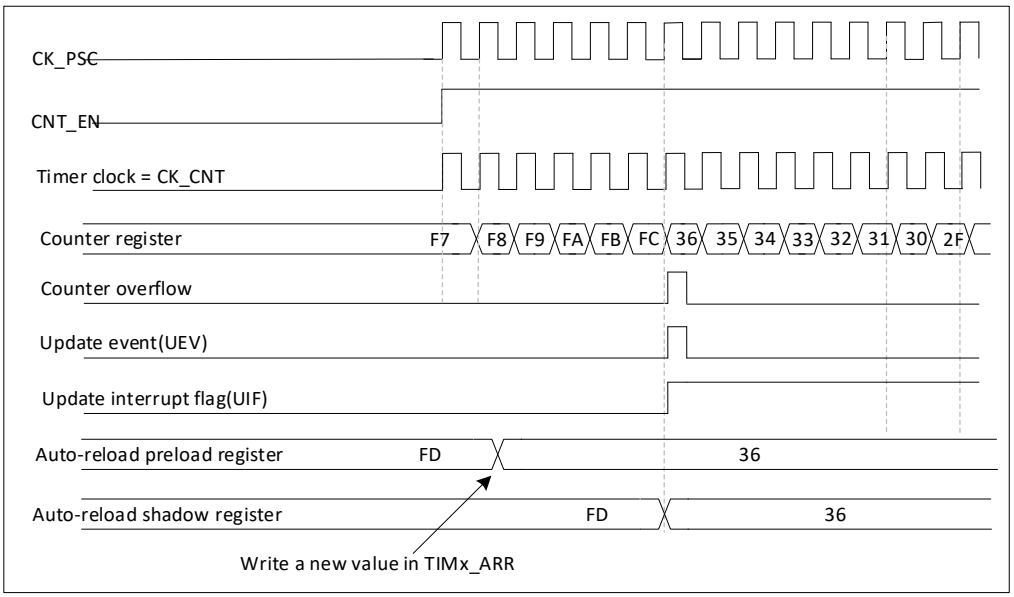

Figure 17-20 Counter timing diagram, Update event with ARPE = 1 (counter overflow)

17.3.3. Repetition counter

Time-base unit describes how the update event (UEV) is generated with respect to the counter overflows/underflows. It is actually generated only when the repetition counter has reached zero. This can be useful when generating PWM signals.

This means that data are transferred from the preload registers to the shadow registers (TIMx_ARR auto-reload register, TIMx_PSC prescaler register, but also TIMx_CCRx capture/compare registers in compare mode) every N+1 counter overflows or underflows,where N is the value in the TIMx_RCR repetition counter register.

The repetition counter is decremented:

- At each counter overflow in upcounting mode.
- At each counter underflow in downcounting mode.
- At each counter overflow and at each counter underflow in center-aligned mode. Although this limits the maximum number of repetition to 128 PWM cycles, it makes it possible to update the duty cycle twice per PWM period. When refreshing compare registers only once per PWM period in center-aligned mode, maximum resolution is 2xTck, due to the symmetry of the pattern.

The repetition counter is an auto-reload type, the repetition rate is maintained as defined by the TIMx_RCR register value. When the update event is generated by software (by setting the UG bit in TIMx_EGR register) or by hardware through the slave mode controller, it occurs immediately whatever the value of the repetition counter is and the repetition counter is reloaded with the content of the TIMx_RCR register.

In center-aligned mode, for odd values of RCR, the update event occurs either on the overflow or on the underflow depending on when the RCR register was written and when the counter was started. If the RCR was written before starting the counter, the UEV occurs on the overflow. If the RCR was written after starting the counter, the UEV occurs on the underflow. For example for RCR = 3, the UEV is generated on each 4th overflow or underflow event depending on when RCR was written.

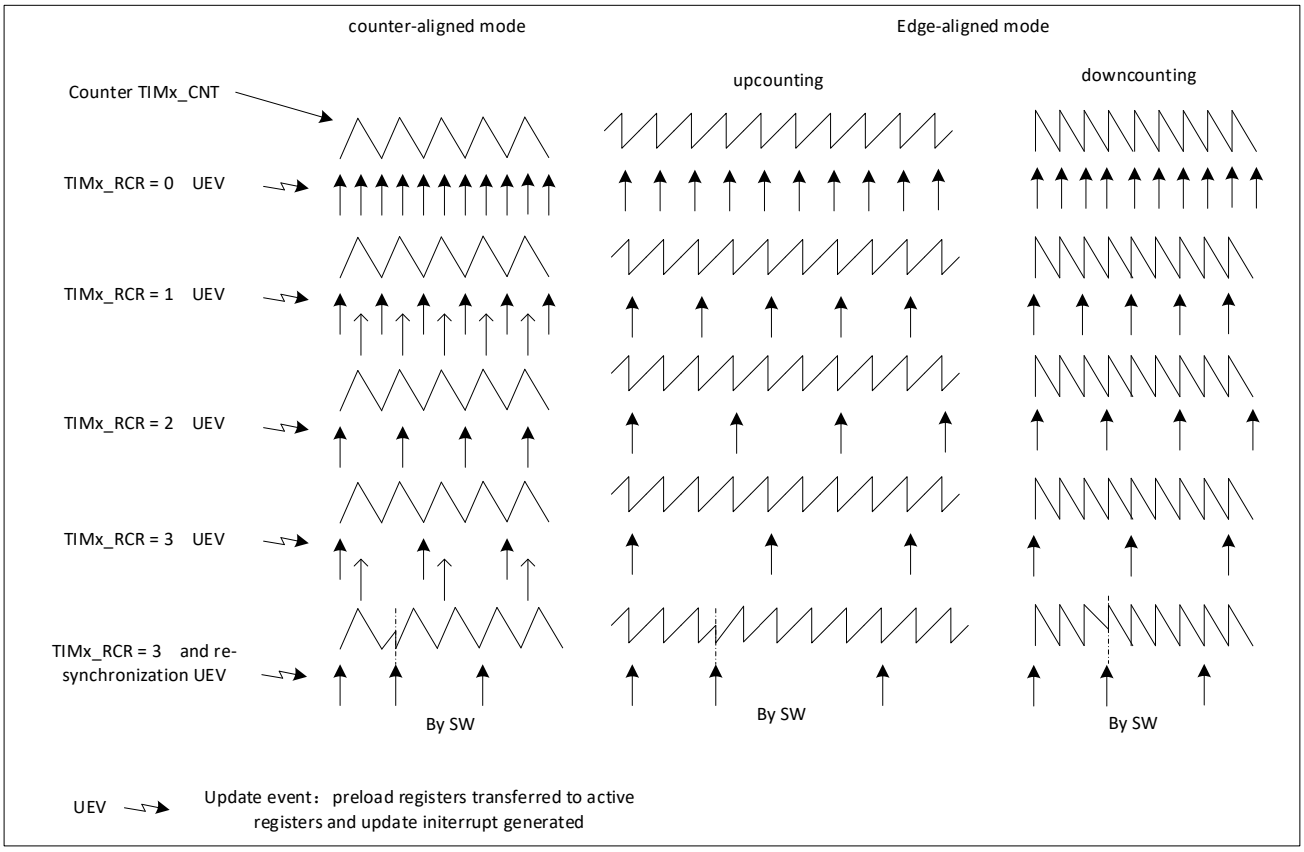

Figure 17-21 Update rate examples depending on mode and TIMx_RCR register settings

17.3.4. Clock selection

The counter clock can be provided by the following clock sources:

- Internal clock (CK_INT)
- External clock mode1: external input pin
- External clock mode2: external trigger input ETR
- Internal trigger inputs (ITRx): using one timer as prescaler for another timer, for example, the user can configure Timer 1 to act as a prescaler for Timer 3.

Internal clock source (CK_INT)

If the slave mode controller is disabled (SMS = 000), then the CEN, DIR (in the TIMx_CR1 register) and UG bits (in the TIMx_EGR register) are actual control bits and can be changed only by software (except UG which remains cleared automatically). As soon as the CEN bit is written to 1, the prescaler is clocked by the internal clock CK_INT.

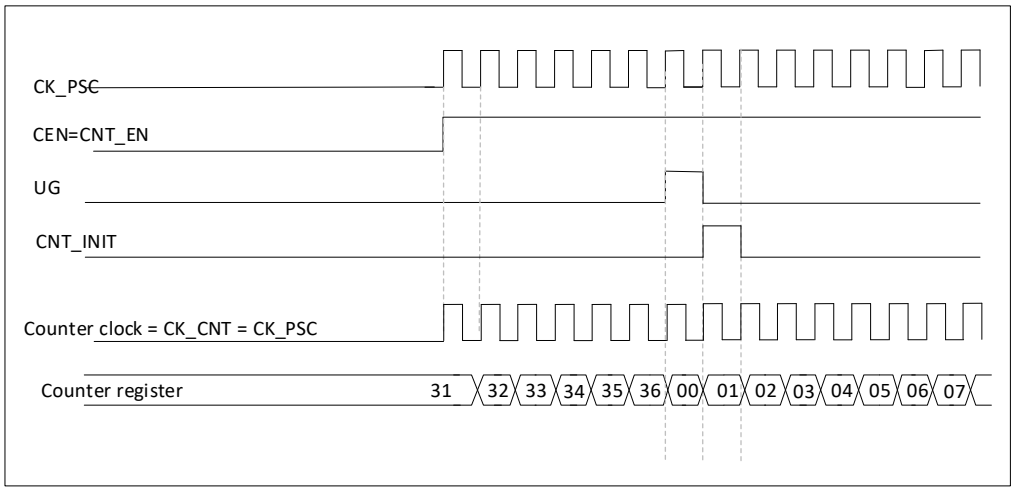

Figure 17-22 Control circuit in normal mode, internal clock divided by 1

External clock source mode 1

This mode is selected when SMS = 111 in the TIMx_SMCR register. The counter can count at each rising or falling edge on a selected input.

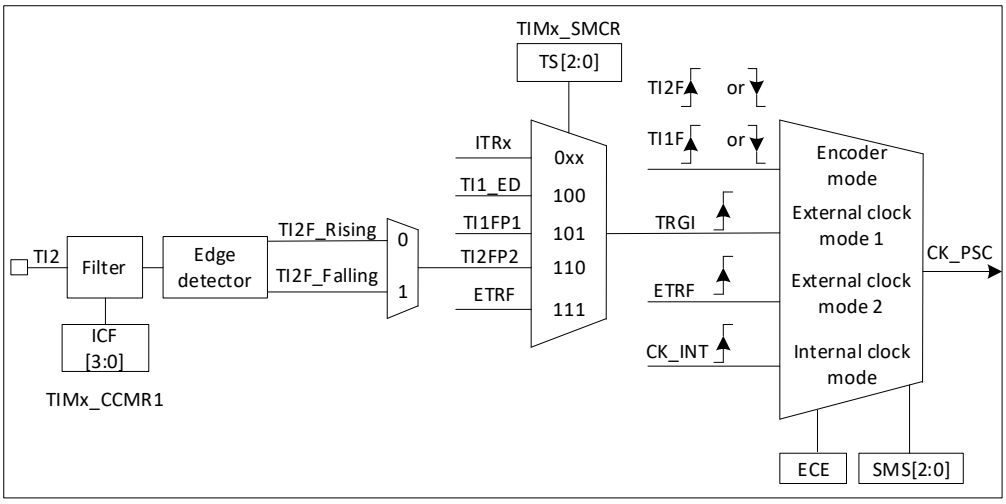

Figure 17-23 TI2 external clock connection example

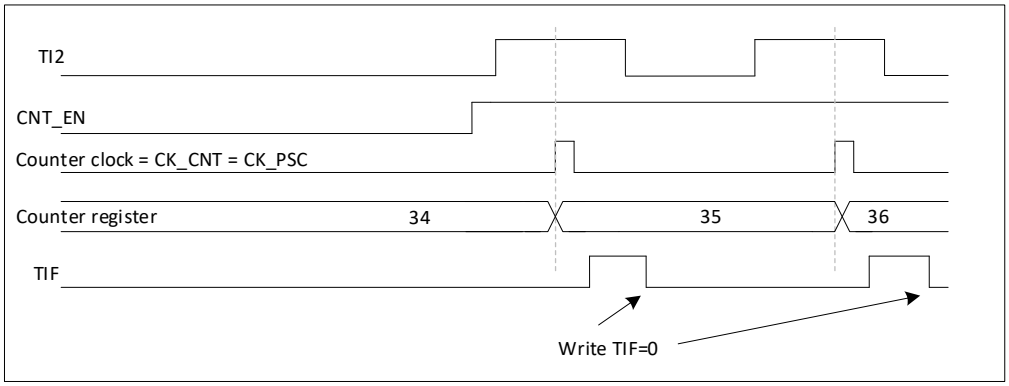

Figure 17-24 Control circuit in external clock mode 1

External clock source mode 2

This mode is selected by writing ECE = 1 in the TIMx_SMCR register. The counter can count at each rising or falling edge on the external trigger input ETR.

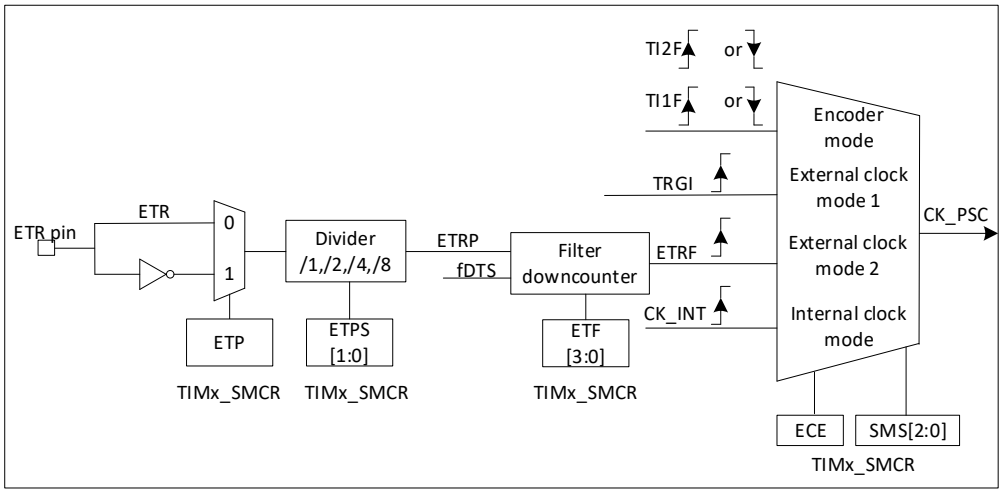

Figure 17-25 External trigger input block

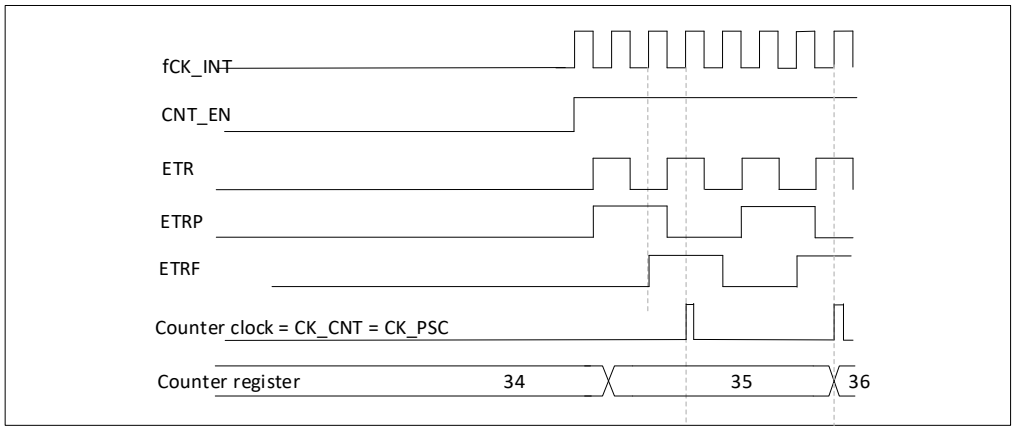

Figure 17-26 Control circuit in external clock mode 2

17.3.5. Capture/compare channels

Each Capture/Compare channel is built around a capture/compare register (including a shadow register), a input stage for capture (with digital filter, multiplexing and prescaler) and an output stage (with comparator and output control).

The input stage samples the corresponding TIx input to generate a filtered signal TIxF. Then, an edge detector with polarity selection generates a signal (TIxFPx) which can be used as trigger input by the slave mode controller or as the capture command. It is prescaled before the capture register (ICxPS).

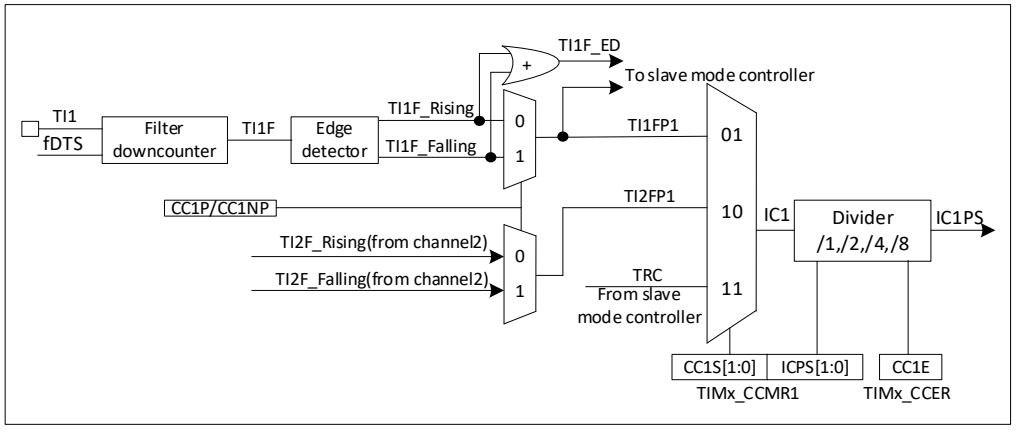

Figure 17-27 Capture/compare channel (example: channel 1 input stage)

The output stage generates an intermediate waveform that is then used for reference: OCxRef (active high). The polarity acts at the end of the chain.

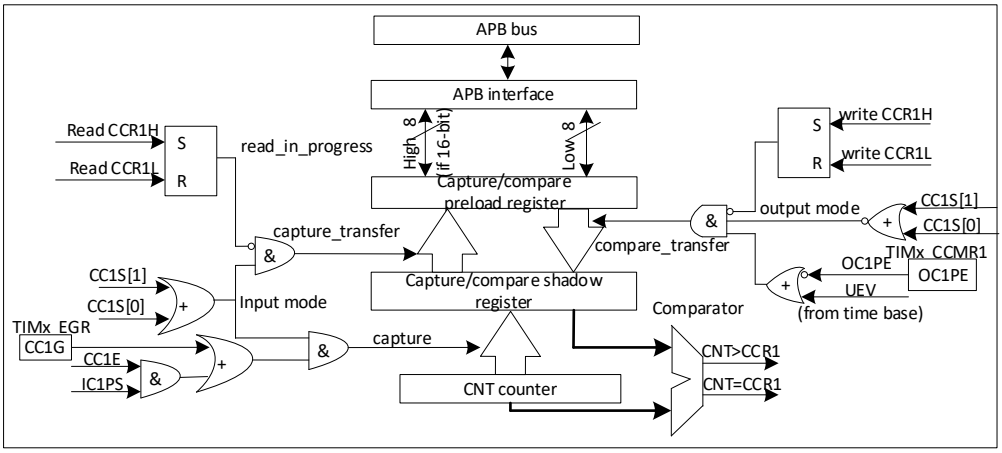

Figure 17-28 Capture/compare channel 1 main circuit

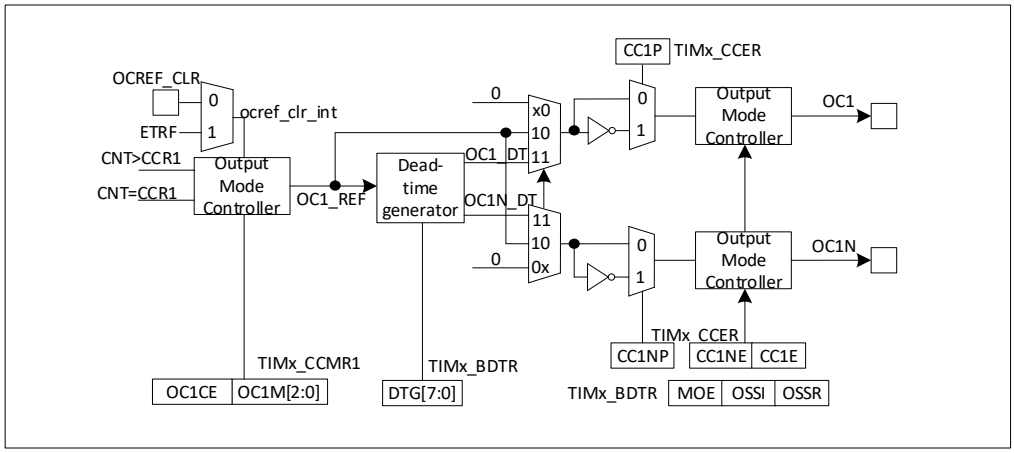

Figure 17-29 Output stage of capture/compare channel (channel 1 to 3)

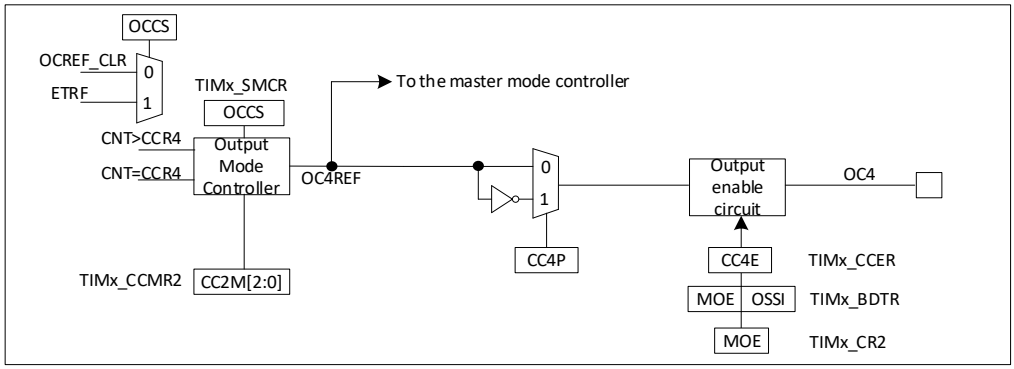

Figure 17-30 Output stage of capture/compare channel (channel 4)

The capture/compare block is made of one preload register and one shadow register. Write and read always access the preload register.

In capture mode, captures are actually done in the shadow register, which is copied into the preload register.

In compare mode, the content of the preload register is copied into the shadow register which is compared to the counter.

17.3.6. Input capture mode

In Input capture mode, the Capture/Compare registers (TIMx_CCRx) are used to latch the value of the counter after a transition detected by the corresponding ICx signal. When a capture occurs, the corresponding CCXIF flag (TIMx_SR register) is set and an interrupt or a DMA request can be sent if they are enabled. If a capture occurs while the CCxIF flag was already high, then the over-capture flag CCxOF (TIMx_SR register) is set. CCxIF can be cleared by software by writing it to '0' or by reading the captured data stored in the TIMx_CCRx register. CCxOF is cleared when written to '0'.

The following example shows how to capture the counter value in TIMx_CCR1 when TI1 input rises. To do this, use the following procedure:

- Select the active input: TIMx_CCR1 must be linked to the TI1 input, so write the CC1S bits to 01 in the TIMx_CCMR1 register. As soon as CC1S becomes different from 00, the channel is configured in input and the TIMx_CCR1 register becomes read-only.
- Program the needed input filter duration with respect to the signal connected to the timer (by programming ICxF bits in the TIMx_CCMRx register if the input is a TIx input). Let's imagine that, when toggling, the input signal is not stable during at must five internal clock cycles. We must program a filter duration longer than these five clockcycles. We can validate a transition on TI1 when 8 consecutive samples with the new level have been detected (sampled at fDTS frequency). Then write IC1F bits to 0011 in the TIMx_CCMR1 register.
- Select the edge of the active transition on the TI1 channel by writing CC1P bit to 0 in the TIMx_CCER register (rising edge in this case).
- Program the input prescaler. In our example, we wish the capture to be performed at each valid transition, so the prescaler is disabled (write IC1PS bits to '00' in the TIMx_CCMR1 register).
- Enable capture from the counter into the capture register by setting the CC1E bit in the TIMx_CCER register.
- If needed, enable the related interrupt request by setting the CC1IE bit in the TIMx_DIER register, and/or the DMA request by setting the CC1DE bit in the TIMx_DIER register.

When an input capture occurs:

- The TIMx_CCR1 register gets the value of the counter on the active transition.
- CC1IF flag is set (interrupt flag). CC1OF is also set if at least two consecutive captures occurred whereas the flag was not cleared.
- An interrupt is generated depending on the CC1IE bit.
- A DMA request is generated depending on the CC1DE bit.

In order to handle the overcapture, it is recommended to read the data before the overcapture flag. This is to avoid missing an overcapture which could happen after reading the flag and before reading the data.

Note: IC interrupt and/or DMA requests can be generated by software by setting the corresponding CCxG bit in the TIMx_EGR register.

17.3.7. PWM input mode

This mode is a particular case of input capture mode. The procedure is the same except:

- Two Icx signals are mapped on the same Tix input.
- The 2 Icx signals are active on edges with opposite polarity.
- One of the two TixFP signals is selected as trigger input and the slave mode controller is configured in reset

mode.

For example, user can measure the period (in TIMx_CCR1 register) and the duty cycle (in TIMx_CCR2 register) of the PWM applied on TI1 using the following procedure (depending on CK_INT frequency and prescaler value):

- Select the active input for TIMx_CCR1: write the CC1S bits to 01 in the TIMx_CCMR1 register (TI1 selected).
- Select the active polarity for TI1FP1 (used both for capture in TIMx_CCR1 and counter clear): write the CC1P bit to '0' (active on rising edge).
- Select the active input for TIMx_CCR2: write the CC2S bits to 10 in the TIMx_CCMR1 register (TI1 selected).
- Select the active polarity for TI1FP2 (used for capture in TIMx_CCR2): write the CC2P bit to '1' (active on falling edge).
- Select the valid trigger input: write the TS bits to 101 in the TIMx_SMCR register (TI1FP1 selected).
- Configure the slave mode controller in reset mode: write the SMS bits to 100 in the TIMx_SMCR register.
- Enable the captures: write the CC1E and CC2E bits to '1' in the TIMx_CCER register.

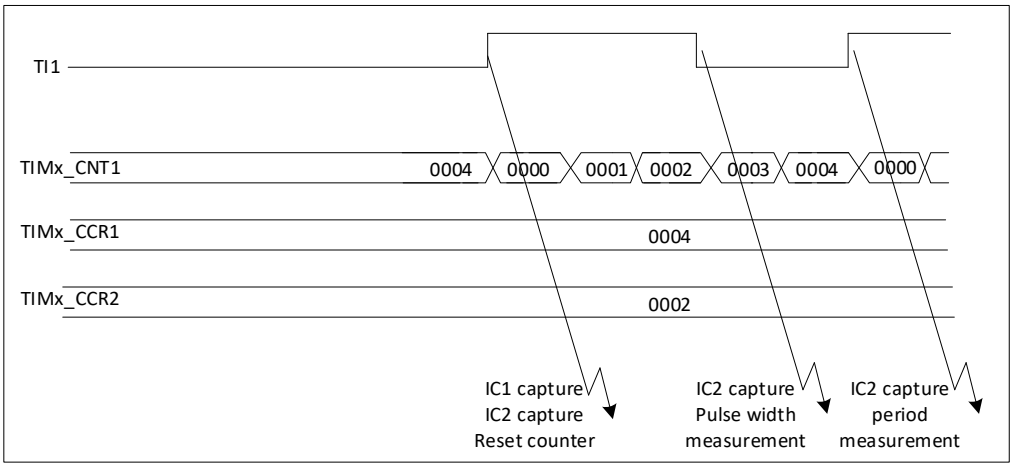

Figure 17-31 PWM input mode timing

17.3.8. Forced output mode

In output mode (CCxS bits = 00 in the TIMx_CCMRx register), each output compare signal (OCxREF and then OCx/OCxN) can be forced to active or inactive level directly by software, independently of any comparison between the output compare register and the counter.

To force an output compare signal (OCXREF/OCx) to its active level, the user just needs to write 101 in the OCxM bits in the corresponding TIMx_CCMRx register. Thus OCXREF is forced high (OCxREF is always active high) and OCx get opposite value to CCxP polarity bit.

For example: CCxP = 0 (OCx active high) = > OCx is forced to high level. The OCxREF signal can be forced low by writing the OCxM bits to 100 in the TIMx_CCMRx register.

The comparison between the TIMx_CCRx shadow register and the counter is still performed and allows the flag to be set. Interrupt and DMA requests can be sent accordingly. This is described in the output compare mode section below.

17.3.9. Output compare mode

This function is used to control an output waveform or indicating when a period of time has elapsed. When a match is found between the capture/compare register and the counter, the output compare function:

- Assigns the corresponding output pin to a programmable value defined by the output compare mode (OCxM

bits in the TIMx_CCMRx register) and the output polarity (CCxP bit in the TIMx_CCER register). The output pin can keep its level (OCXM = 000), be set active (OCxM = 001), be set inactive (OCxM = 010) or can toggle (OCxM = 011) on match.

- Sets a flag in the interrupt status register (CCxIF bit in the TIMx_SR register).
- Generates an interrupt if the corresponding interrupt mask is set (CCXIE bit in the TIMx_DIER register).
- Sends a DMA request if the corresponding enable bit is set (CCxDE bit in the TIMx_DIER register, CCDS bit in the TIMx_CR2 register for the DMA request selection).

The TIMx_CCRx registers can be programmed with or without preload registers using the OCxPE bit in the TIMx_CCMRx register.

In output compare mode, the update event UEV has no effect on OCxREF and OCx output.

The timing resolution is one count of the counter. Output compare mode can also be used to output a single pulse (in One Pulse mode).

Procedure:

1. Select the counter clock (internal, external, prescaler).
2. Write the desired data in the TIMx_ARR and TIMx_CCRx registers.
3. Set the CCxIE bit if an interrupt request is to be generated.
4. Select the output mode. For example:

– Write OCxM = 011 to toggle OCx output pin when CNT matches CCRx

– Write OCxPE = 0 to disable preload register

– Write CCxP = 0 to select active high polarity

– Write CCxE = 1 to enable the output

5. Enable the counter by setting the CEN bit in the TIMx_CR1 register.

The TIMx_CCRx register can be updated at any time by software to control the output waveform, provided that the preload register is not enabled (OCxPE = ’0’, else TIMx_CCRxshadow register is updated only at the next update event UEV). An example is given in Figure 17-32.

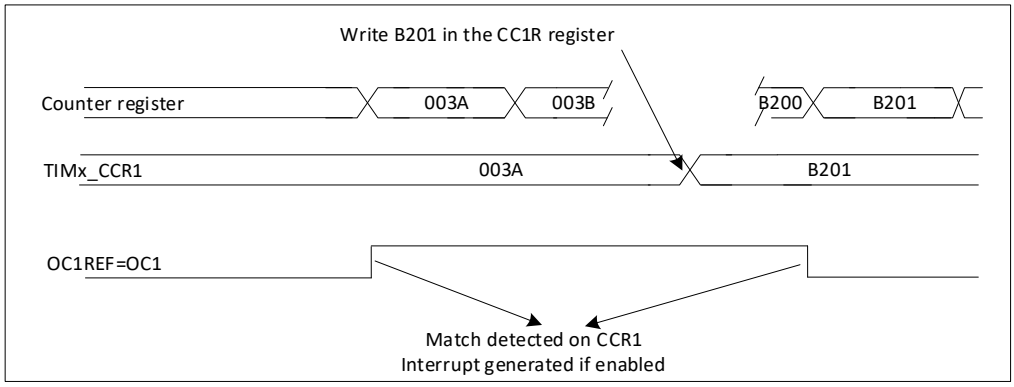

Figure 17-33 Output compare mode, toggle on OC1.

17.3.10. PWM mode

Pulse Width Modulation mode allows generating a signal with a frequency determined by the value of the TIMx_ARR register and a duty cycle determined by the value of the TIMx_CCRx register.

The PWM mode can be selected independently on each channel (one PWM per OCx output) by writing ‘110’ (PWM mode 1) or ‘111’ (PWM mode 2) in the OCxM bits in the TIMx_CCMRx register. The corresponding

preload register must be enabled by setting the OCxPE bit in the TIMx_CCMRx register, and eventually the auto-reload preload register (in upcounting or center-aligned modes) by setting the ARPE bit in the TIMx_CR1 register.

As the preload registers are transferred to the shadow registers only when an update event occurs, before starting the counter, the user must initialize all the registers by setting the UG bit in the TIMx_EGR register.

OCx polarity is software programmable using the CCxP bit in the TIMx_CCER register. It can be programmed as active high or active low. OCx output is enabled by a combination of the CCxE, CCxNE, MOE, OSSI and OSSR bits (TIMx_CCER and TIMx_BDTR registers). Refer to the TIMx_CCER register description for more details.

In PWM mode (1 or 2), TIMx_CNT and TIMx_CCRx are always compared to determine whether TIMx_CCRx ≤ TIMx_CNT or TIMx_CNT ≤ TIMx_CCRx (depending on the direction of the counter).

The timer is able to generate PWM in edge-aligned mode or center-aligned mode depending on the CMS bits in the TIMx_CR1 register.

PWM edge-aligned mode

- **Upcounting configuration**

Upcounting is active when the DIR bit in the TIMx_CR1 register is low. Refer to Upcounting mode. Upcounting is active when the DIR bit in the TIMx_CR1 register is low. Refer to Upcounting mode.

In the following example, we consider PWM mode 1. The reference PWM signal OCxREF is high as long as TIMx_CNT < TIMx_CCRx else it becomes low. If the compare value in TIMx_CCRx is greater than the auto-reload value (in TIMx_ARR) then OCxREF is held at ‘1’. If the compare value is 0 then OCxRef is held at ‘0’. Figure 17-34 shows some edge-aligned PWM waveforms in an example where TIMx_ARR = 8.

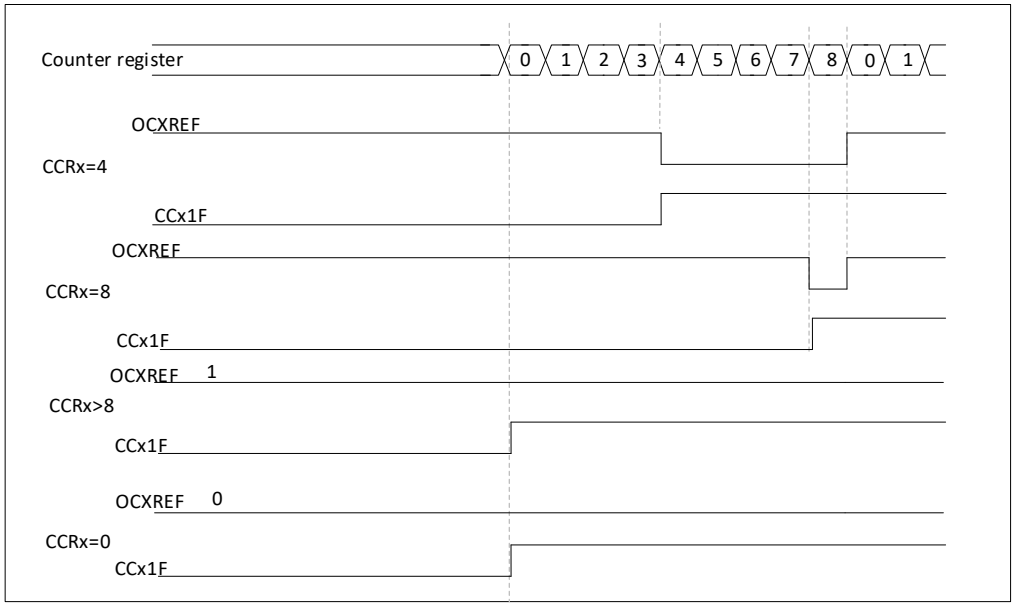

Figure 17-35 Edge-aligned PWM waveforms (ARR = 8)

- **Downcounting configuration**

Downcounting is active when DIR bit in TIMx_CR1 register is high.

In PWM mode 1, the reference signal OCxRef is low as long as TIMx_CNT > TIMx_CCRx else it becomes high. If the compare value in TIMx_CCRx is greater than the auto-reload value in TIMx_ARR, then OCxREF is held at ‘1’. 0% PWM is not possible in this mode.

PWM center-aligned mode

Center-aligned mode is active when the CMS bits in TIMx_CR1 register are different from '00' (all the remaining configurations having the same effect on the OCxRef/OCx signals).

The compare flag is set when the counter counts up, when it counts down or both when it counts up and down depending on the CMS bits configuration. The direction bit (DIR) in the TIMx_CR1 register is updated by hardware and must not be changed by software. Refer to Center-aligned mode (up/down counting).

- TIMx_ARR = 8
- PWM mode is the PWM mode 1,
- The flag is set when the counter counts down corresponding to the center-aligned mode 1 selected for CMS = 01 in TIMx_CR1 register.

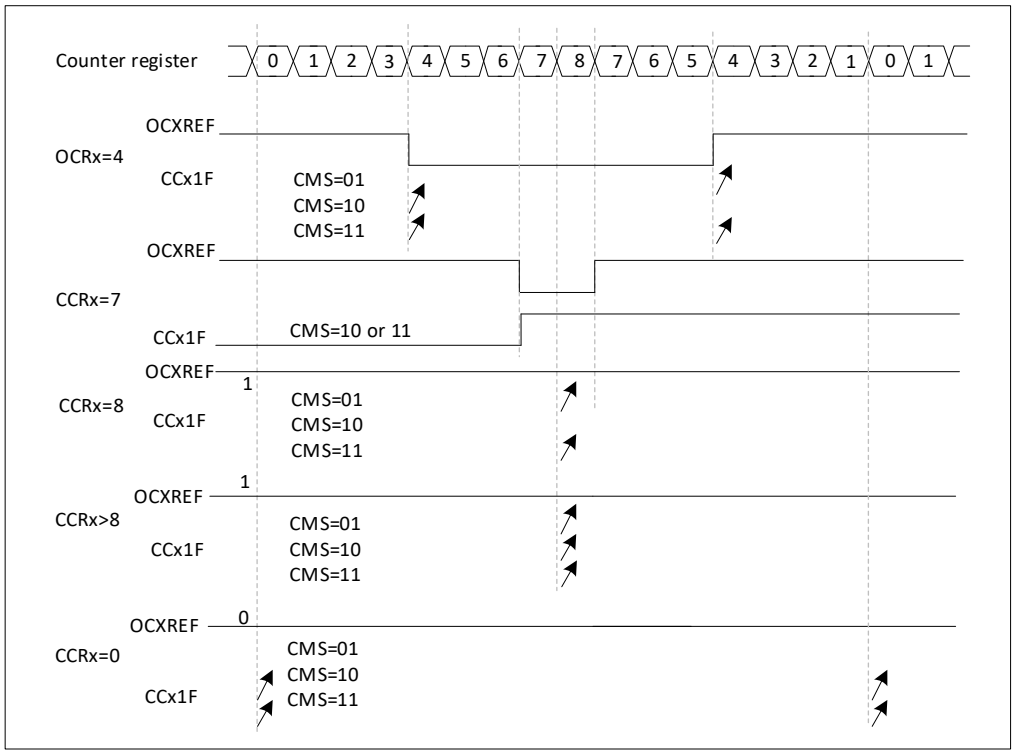

Figure 17-36 Center-aligned PWM waveforms (ARR = 8)

Hints on using center-aligned mode:

- When starting in center-aligned mode, the current up-down configuration is used. It means that the counter counts up or down depending on the value written in the DIR bit in the TIMx_CR1 register. Moreover, the DIR and CMS bits must not be changed at the same time by the software.
- Writing to the counter while running in center-aligned mode is not recommended as it can lead to unexpected results. In particular: – The direction is not updated if the user writes a value in the counter greater than the auto-reload value (TIMx_CNT > TIMx_ARR). For example, if the counter was counting up, it will continue to count up. – The direction is updated if the user writes 0 or write the TIMx_ARR value in the counter but no Update Event UEV is generated.
- The safest way to use center-aligned mode is to generate an update by software (setting the UG bit in the TIMx_EGR register) just before starting the counter and not to write the counter while it is running.

17.3.11. Complementary outputs and dead-time insertion

The advanced-control timers (TIM1) can output two complementary signals and manage the switching-off and the switching-on instants of the outputs. This time is generally known as dead-time and it has to be adjust it depending on the devices connected to the outputs and their characteristics (intrinsic delays of level-shifters, delays due to power switches).

User can select the polarity of the outputs (main output OCx or complementary OCxN) independently for each output. This is done by writing to the CCxP and CCxNP bits in the TIMx_CCER register.

The complementary signals OCx and OCxN are activated by a combination of several control bits: the CCxE and CCxNE bits in the TIMx_CCER register and the MOE, OISx, OISxN, OSSI and OSSR bits in the TIMx_BDTR and TIMx_CR2 registers. Refer to Table 17-3 for more details. In particular, the dead-time is activated when switching to the IDLE state (MOE falling down to 0).

Dead-time insertion is enabled by setting both CCxE and CCxNE bits, and the MOE bit if the break circuit is present. DTG[7:0] bits of the TIMx_BDTR register are used to control the dead-time generation for all channels. From a reference waveform OCxREF, it generates 2 outputs OCx and OCxN. If OCx and OCxN are active high:

- The OCx output signal is the same as the reference signal except for the rising edge, which is delayed relative to the reference rising edge.
- The OCxN output signal is the opposite of the reference signal except for the rising edge, which is delayed relative to the reference falling edge.

If the delay is greater than the width of the active output (OCx or OCxN) then the corresponding pulse is not generated.

The following figures show the relationships between the output signals of the dead-time generator and the reference signal OCxREF. (we suppose CCxP = 0, CCxNP = 0, MOE = 1, CCxE = 1 and CCxNE = 1 in these examples).

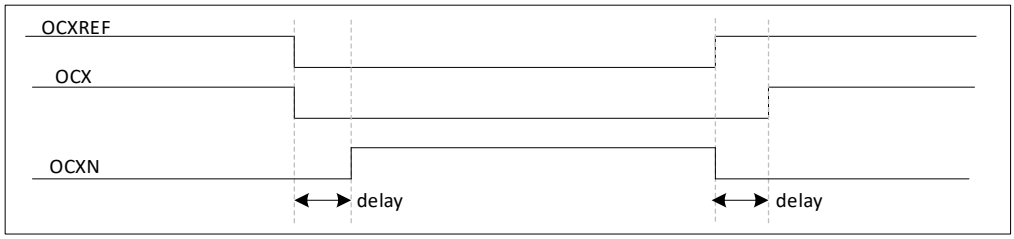

Figure 17-37 Complementary output with dead-time insertion

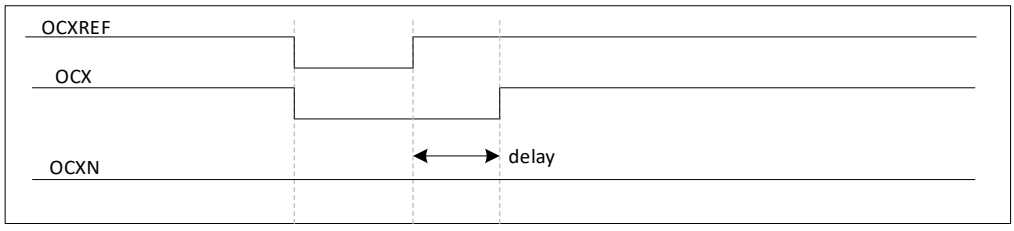

Figure 17-38 Dead-time waveforms with delay greater than the negative pulse.

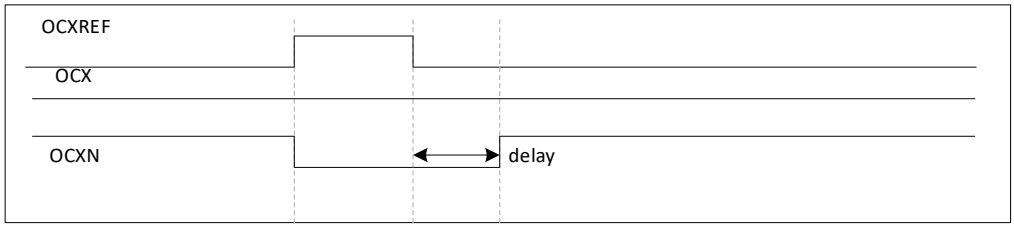

Figure 17-39 Dead-time waveforms with delay greater than the positive pulse.

The dead-time delay is the same for each of the channels and is programmable with the DTG bits in the TIMx_BDTR register.

Re-directing OCxREF to OCx or OCxN

In output mode (forced, output compare or PWM), OCxREF can be re-directed to the OCx output or to OCxN output by configuring the CCxE and CCxNE bits in the TIMx_CCER register.

This allows the user to send a specific waveform (such as PWM or static active level) on one output while the complementary remains at its inactive level. Other possibilities are to have both outputs at inactive level or both outputs active and complementary with dead-time.

Note: When only OCxN is enabled (CCxE = 0, CCxNE = 1), it is not complemented and becomes active as soon as OCxREF is high. For example, if CCxNP = 0 then OCxN = OCxRef. On the other hand, when both OCx and OCxN are enabled (CCxE = CCxNE = 1) OCx becomes active when OCxREF is high whereas OCxN is complemented and becomes active when OCxREF is low.

17.3.12. Using the break function

When using the break function, the output enable signals and inactive levels are modified according to additional control bits (MOE, OSSI and OSSR bits in the TIMx_BDTR register, OISx and OISxN bits in the TIMx_CR2 register). In any case, the OCx and OCxN outputs cannot be set both to active level at a given time. Refer to Table 17-3 for more details.

The brake source can be either the brake input pin, or the following internal sources:

- Output from CPU LOCKUP
- Ouput from PVD
- Clock failure events generated by the Clock Security System (CSS)
- Output from comparator

After reset, the break circuit is disabled and the MOE bit is low. User can enable the break function by setting the BKE bit in the TIMx_BDTR register. The break input polarity can be selected by configuring the BKP bit in the same register. BKE and BKP can be modified at the same time. When the BKE and BKP bits are written, a delay of 1 APB clock cycle is applied before the writing is effective. Consequently, it is necessary to wait 1 APB clock period to correctly read back the bit after the write operation.

Because MOE falling edge can be asynchronous, a resynchronization circuit has been inserted between the actual signal (acting on the outputs) and the synchronous control bit (accessed in the TIMx_BDTR register). It results in some delays between the asynchronous and the synchronous signals. In particular, if MOE is written to 1 whereas it was low, a delay (dummy instruction) must be inserted before reading it correctly. This is because the user writes an asynchronous signal, but reads a synchronous signal.

When a break occurs (selected level on the break input):

- he MOE bit is cleared asynchronously, putting the outputs in inactive state, idle state or in reset state (selected by the OSSI bit). This feature functions even if the MCU oscillator is off.
- Each output channel is driven with the level programmed in the OISx bit in the TIMx_CR2 register as soon as MOE = 0. If OSSI = 0 then the timer releases the enable output else the enable output remains high.
- When complementary outputs are used:

- The outputs are first put in reset state inactive state (depending on the polarity). This is done asynchronously so that it works even if no clock is provided to the timer.
- If the timer clock is still present, then the dead-time generator is reactivated in order to drive the outputs with the level programmed in the OISx and OISxN bits after a dead-time. Even in this case, OCx and OCxN cannot be driven to their active level together. Note that because of the resynchronization on MOE, the dead-time duration is a bit longer than usual (around 2 ck_tim clock cycles).
- If OSSI = 0 then the timer releases the enable outputs else the enable outputs remain or become high as soon as one of the CCxE or CCxNE bits is high.

- The break status flag (BIF bit in the TIMx_SR register) is set. An interrupt can be generated if the BIE bit in the TIMx_DIER register is set.
- If the AOE bit in the TIMx_BDTR register is set, the MOE bit is automatically set again at the next update event UEV. This can be used to perform a regulation, for instance. Else, MOE remains low until it is written to '1' again. In this case, it can be used for security and the break input can be connected to an alarm from power drivers, thermal sensors or any security components.

Note: The break inputs is acting on level. Thus, the MOE cannot be set while the break input is active (neither automatically nor by software). In the meantime, the status flag BIF cannot be cleared.

The break can be generated by the BRK input which has a programmable polarity and an enable bit BKE in the TIMx_BDTR register.

By using the BRK input which has a programmable polarity and an enable bit BKE in the TIMx_BDTR register

In addition to the break input and the output management, a write protection has been implemented inside the break circuit to safeguard the application. It allows freezing the configuration of several parameters (dead-time duration, OCx/OCxN polarities and state when disabled, OCxM configurations, break enable and polarity). The user can choose from three levels of protection selected by the LOCK bits in the TIMx_BDTR register. The LOCK bits can be written only once after an MCU reset.

The following figure shows the example of the output in response to a break.

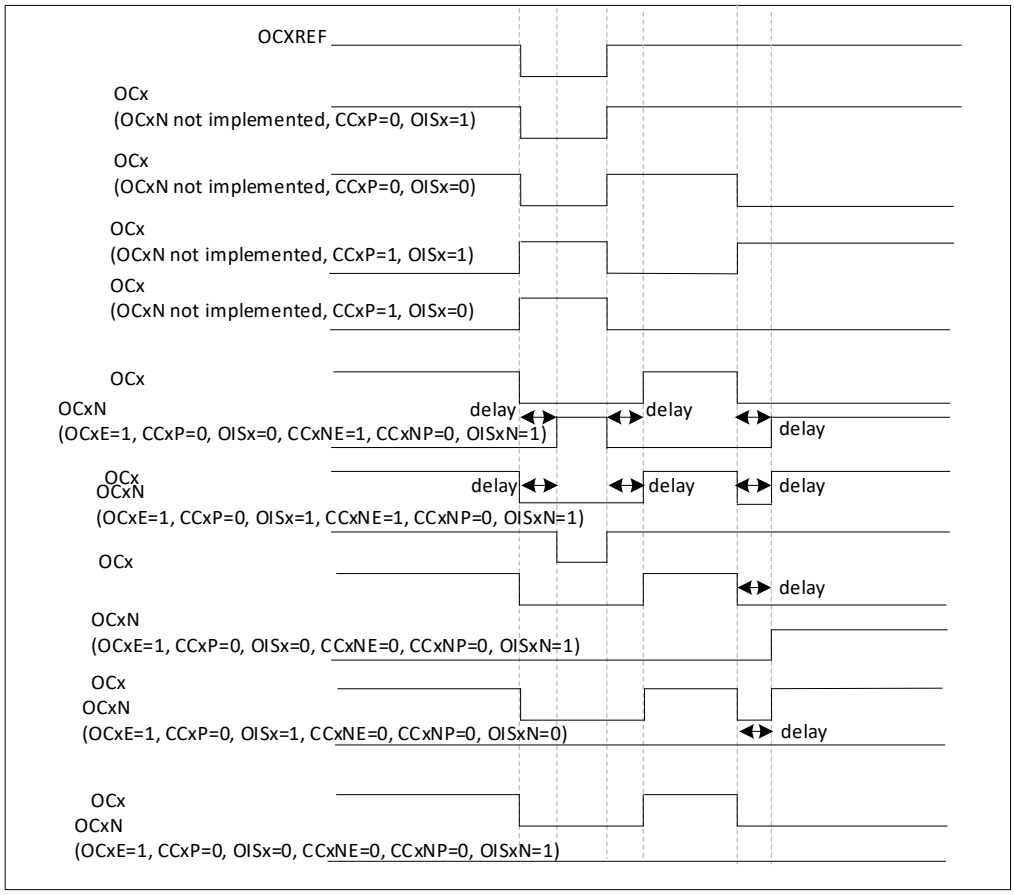

Figure 17-40 Output behavior in response to a break

17.3.13. Clearing the OCxREF signal on an external event

The OCxREF signal for a given channel can be driven Low by applying a High level to the ETRF input (OCxCE enable bit of the corresponding TIMx_CCMRx register set to ‘1’). The OCxREF signal remains Low until the next update event of UEV occurs.

This function can only be used in output compare and PWM modes, and does not work in forced mode.

For example, the ETR signal can be connected to the output of a comparator to be used for current handling. In this case, the ETR must be configured as follow:

1. The External Trigger Prescaler should be kept off: bits ETPS[1:0] of the TIMx_SMCR register set to ‘00’.
2. The external clock mode 2 must be disabled: bit ECE of the TIMx_SMCR register set to ‘0’.
3. The External Trigger Polarity (ETP) and the External Trigger Filter (ETF) can be configured according to the user needs.

The Figure below shows the behavior of the OCxREF signal when the ETRF Input becomes High, for both values of the enable bit OCxCE. In this example, the timer TIMx is programmed in PWM mode.

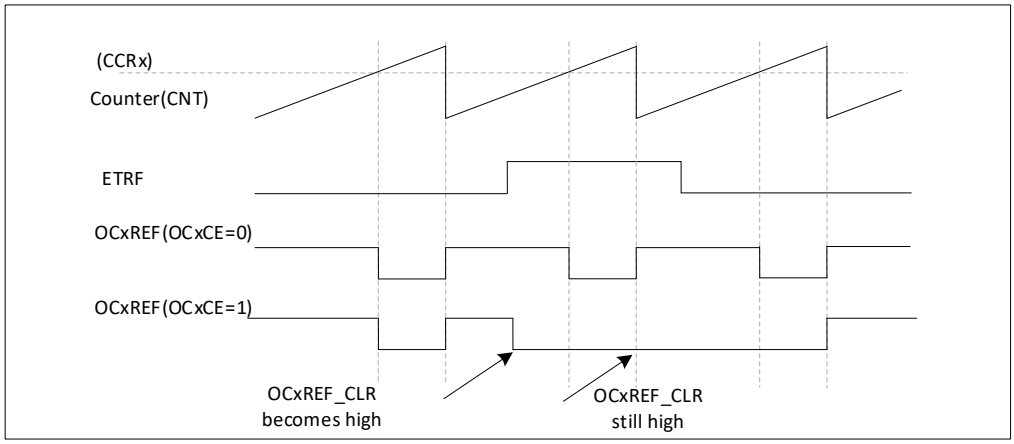

Figure 17-41 Clearing TIM1 OCxREF

17.3.14. 6-step PWM generation

When complementary outputs are used on a channel, preload bits are available on the OCxM, CCxE and CCxNE bits. The preload bits are transferred to the shadow bits at the COM commutation event. The user can thus program in advance the configuration for the next step and change the configuration of all the channels at the same time. COM can be generated by software by setting the COM bit in the TIMx_EGR register or by hardware (on TRGI rising edge).

A flag is set when the COM event occurs (COMIF bit in the TIMx_SR register), which can generate an interrupt (if the COMIE bit is set in the TIMx_DIER register) or a DMA request (if the COMDE bit is set in the TIMx_DIER register).

The figure blow describes the behavior of the OCx and OCxN outputs when a COM event occurs, in 3 different examples of programmed configurations.

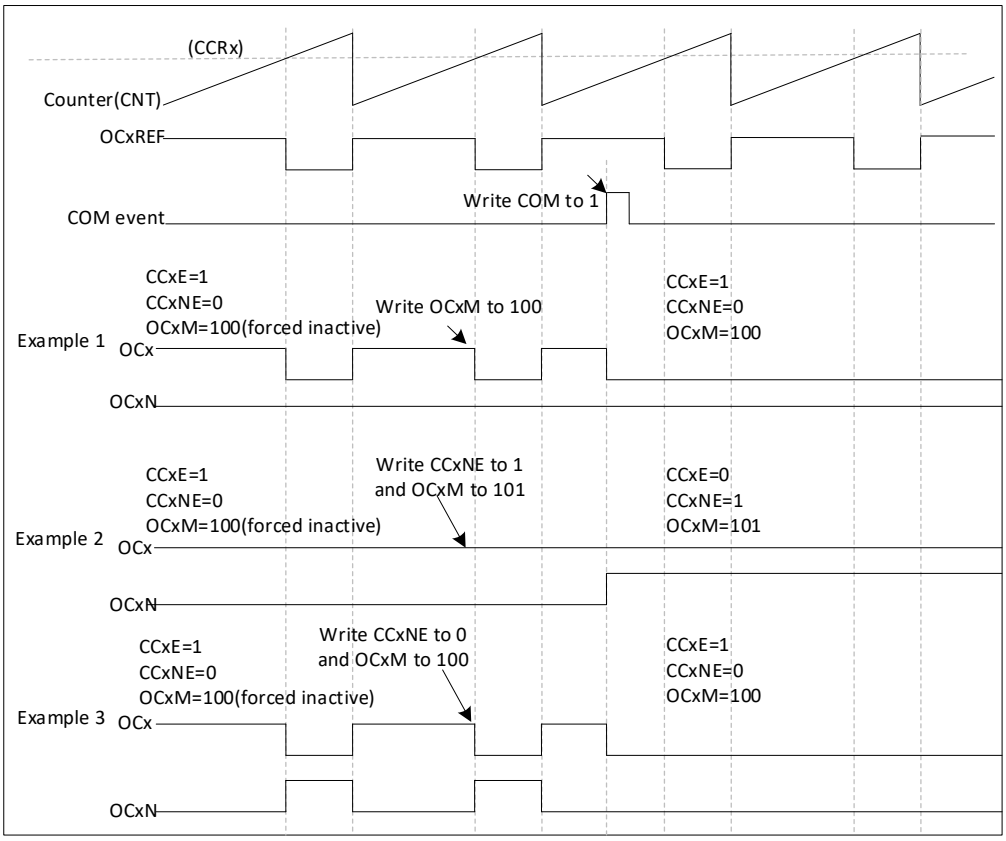

Figure 17-42 6-step generation, COM example (OSSR = 1)

17.3.15. One-pulse mode

One-pulse mode (OPM) is a particular case of the previous modes. It allows the counter to be started in response to a stimulus and to generate a pulse with a programmable length after a programmable delay.

Starting the counter can be controlled through the slave mode controller. Generating the waveform can be done in output compare mode or PWM mode. Select One-pulse mode by setting the OPM bit in the TIMx_CR1 register. This makes the counter stop automatically at the next update event UEV.

A pulse can be correctly generated only if the compare value is different from the counter initial value.

Before starting (when the timer is waiting for the trigger), the configuration must be:

- In upcounting: CNT < CCRx ≤ ARR (in particular, 0 < CCRx)
- In downcounting: CNT > CCRx

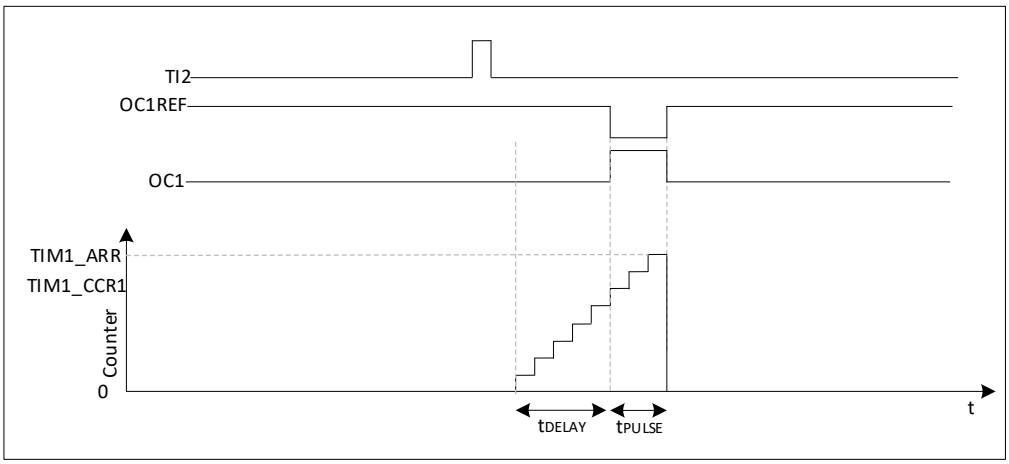

Figure 17-43 Example of one pulse mode

For example the user may want to generate a positive pulse on OC1 with a length of t_{PULSE} and after a delay of t_{DELAY} as soon as a positive edge is detected on the TI2 input pin.

Let's use TI2FP2 as trigger 1:

- Map TI2FP2 to TI2 by writing CC2S = '01' in the TIMx_CCMR1 register.
- TI2FP2 must detect a rising edge, write CC2P = '0' in the TIMx_CCER register.
- Configure TI2FP2 as trigger for the slave mode controller (TRGI) by writing TS = '110' in the TIMx_SMCR register.
- TI2FP2 is used to start the counter by writing SMS to '110' in the TIMx_SMCR register (trigger mode).

The OPM waveform is defined by writing the compare registers (taking into account the clock frequency and the counter prescaler).

- The tDELAY is defined by the value written in the TIMx_CCR1 register.
- The tPULSE is defined by the difference between the auto-reload value and the compare value (TIMx_ARR-TIMx_CCR1).
- When the user to build a waveform with a transition from '0' to '1' when a compare match oc-curs and a transition from '1' to '0' when the counter reaches the auto-reload value. To do this, enable PWM mode 2 by writing OC1M = 111 in the TIMx_CCMR1 register. The user can optionally enable the preload registers by writing OC1PE = '1' in the TIMx_CCMR1 register and ARPE in the TIMx_CR1 register. In this case the compare value must be written in the TIMx_CCR1 register, the auto-reload value in the TIMx_ARR register, generate an update by setting the UG bit and wait for external trigger event on TI2. CC1P is written to '0' in this example.

In this example, the DIR and CMS bits in the TIMx_CR1 register should be low.

The user only wants one pulse (Single mode), so '1' must be written in the OPM bit in the TIMx_CR1 register to stop the counter at the next update event (when the counter rolls over from the auto-reload value back to 0). When OPM bit in the TIMx_CR1 register is set to '0', so the Repetitive Mode is selected.

Particular case: OCx fast enable:

In One-pulse mode, the edge detection on TIx input set the CEN bit which enables the counter. Then the comparison between the counter and the compare value makes the output toggle. But several clock cycles are needed for these operations and it limits the minimum delay tDELAY.

If the user wants to output a waveform with the minimum delay, the OCxFE bit in the TIMx_CCMRx register must be set. Then OCxRef (and OCx) are forced in response to the stimulus, without taking in account the comparison. Its new level is the same as if a compare match had occurred. OCxFE acts only if the channel is configured in PWM1 or PWM2 mode.

17.3.16. Encoder interface mode

To select Encoder Interface mode write SMS = '001' in the TIMx_SMCR register if the counter is counting on TI2 edges only, SMS = '010' if it is counting on TI1 edges only and SMS = '011' if it is counting on both TI1 and TI2 edges.

Select the TI1 and TI2 polarity by programming the CC1P and CC2P bits in the TIMx_CCER register. When needed, the user can program the input filter as well.

The two inputs TI1 and TI2 are used to interface to an incremental encoder. Refer to Table 17-1. The counter is clocked by each valid transition on TI1FP1 or TI2FP2 (TI1 and TI2 after input filter and polarity selection, TI1FP1 = TI1 if not filtered and not inverted,TI2FP2 = TI2 if not filtered and not inverted) assuming that it is enabled (CEN bit in TIMx_CR1 register written to '1'). The sequence of transitions of the two inputs is evaluated and generates count pulses as well as the direction signal. Depending on the sequence the counter counts up or down, the DIR bit in the TIMx_CR1 register is modified by hardware accordingly. The DIR bit is calculated at each transition on any input (TI1 or TI2), whatever the counter is counting on TI1 only, TI2 only or both TI1 and TI2.

Encoder interface mode acts simply as an external clock with direction selection. This means that the counter just counts continuously between 0 and the auto-reload value in the TIMx_ARR register (0 to ARR or ARR down to 0 depending on the direction). So user must configure TIMx_ARR before starting. in the same way, the capture, compare, prescaler, repetition counter, trigger output features continue to work as normal. Encoder mode and External clock mode 2 are not compatible and must not be selected together. In this mode, the counter is modified automatically following the speed and the direction of the incremental encoder and its content, therefore, always represents the encoder's position. The count direction correspond to the rotation direction of the connected sensor. The table below summarizes the possible combinations, assuming TI1 and TI2 do not switch at the same time.

Table 17-1 Counting direction versus encoder signals

Active edge	Level on opposite signal (TI1FP1 for TI2, TI2FP2 for TI1)	TI1FP1 signal		TI2FP2 signal	
		Rising	Falling	Rising	Falling
Counting on TI1 only	High	Down	Up	No count	No count
	Low	Up	Down	No count	No count
Counting on TI2 only	High	No count	No count	Up	Down
	Low	No count	No count	Down	Up
Counting on TI1 and TI2	High	Down	Up	Up	Down
	Low	Up	Down	Down	Up

An external incremental encoder can be connected directly to the MCU without external interface logic. However, comparators are normally be used to convert the encoder's differential outputs to digital signals. This greatly increases noise immunity. The third encoder output which indicate the mechanical zero position, may be connected to an external interrupt input and trigger a counter reset.

The figure below gives an example of counter operation, showing count signal generation and direction control. It also shows how input jitter is compensated where both edges are selected. This might occur if the sensor is positioned near to one of the switching points. For this example we assume that the configuration is the following:

- CC1S = ‘01’(TIMx_CCMR1 register, TI1FP1 mapped on TI1).
- CC2S = ‘01’ (TIMx_CCMR2 register, TI1FP2 mapped on TI2).
- CC1P = ‘0’, and IC1F = ‘0000’ (TIMx_CCER register, TI1FP1 non inverted, TI1FP1 = TI1).
- CC2P = ‘0’, and IC2F = ‘0000’ (TIMx_CCER register, TI1FP2 non-inverted, TI1FP2 = TI2).
- SMS = ‘011’ (TIMx_SMCR register, both inputs are active on both rising and falling edges).
- CEN = ‘1’ (TIMx_CR1 register, counter enabled).

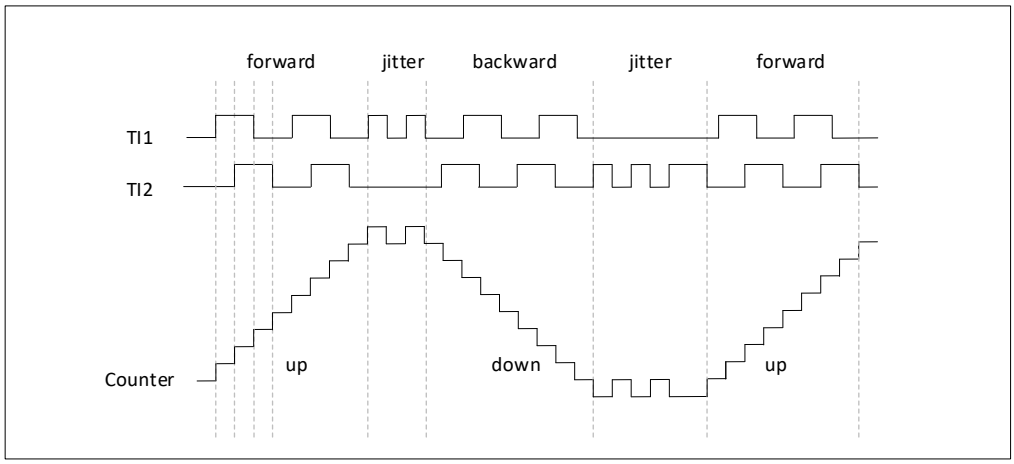

Figure 17-44 Example of counter operation in encoder interface mode.

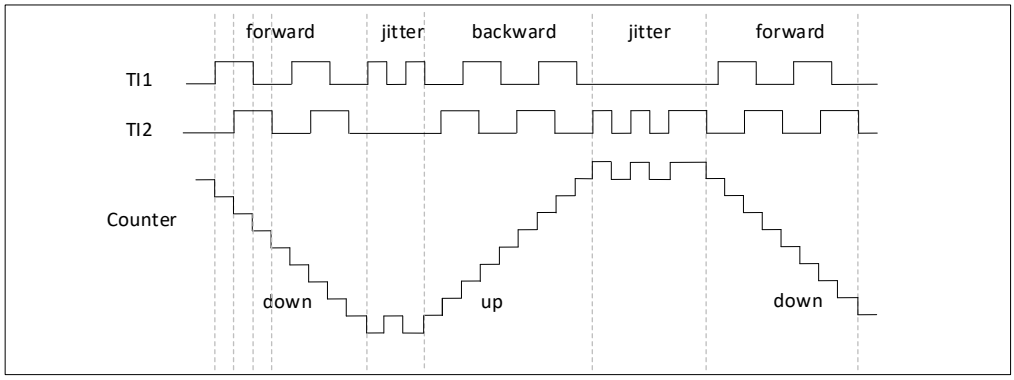

Figure 17-45 Example of encoder interface mode with TI1FP1 polarity inverted.

The timer, when configured in Encoder Interface mode provides information on the sensor’s current position.The user can obtain dynamic information (speed, acceleration, deceleration) by measuring the period between two encoder events using a second timer configured in capture mode. The output of the encoder which indicates the mechanical zero can be used for this purpose. Depending on the time between two events, the counter can also be read at regular times. This can be done by latching the counter value into a third input capture register if available (then the capture signal must be periodic and can be generated by another timer). When available, it is also possible to read its value through a DMA request generated by a real-time clock.

17.3.17. Timer input XOR function

The TI1S bit in the TIMx_CR2 register, allows the input filter of channel 1 to be connected to the output of a XOR gate, combining the three input pins TIMx_CH1, TIMx_CH2 and TIMx_CH3.

The XOR output can be used with all the timer input functions such as trigger or input capture.

17.3.18. Interfacing with Hall sensors

This is done using the advanced-control timers (TIM1) to generate PWM signals to drive the motor and another timer TIMx (TIM3) referred to as “interfacing timer” in Figure 17-44. The “interfacing timer” captures the 3 timer input pins (TIMx_CH1, TIMx_CH2, and TIMx_CH3) connected through a XOR to the TI1 input channel (selected by setting the TI1S bit in the TIMx_CR2 register).

The slave mode controller is configured in reset mode, the slave input is TI1F_ED. Thus, each time one of the 3 inputs toggles, the counter restarts counting from 0. This creates a time base triggered by any change on the Hall inputs.

On the “interfacing timer”, capture/compare channel 1 is configured in capture mode, capture signal is TRC (see Figure 17-27). The captured value, which corresponds to the time elapsed between 2 changes on the inputs, gives information about motor speed.

The “interfacing timer” can be used in output mode to generate a pulse which changes the configuration of the channels of the advanced-control timer (TIM1) (by triggering a COM event). The TIM1 timer is used to generate PWM signals to drive the motor. To do this, the interfacing timer channel must be programmed so that a positive pulse is generated after a programmed delay (in output compare or PWM mode). This pulse is sent to the advanced-control timer (TIM1) through the TRGO output.

Example: the user wants to change the PWM configuration of the advanced-control timer TIM1 after a programmed delay each time a change occurs on the Hall inputs connected to one of the TIMx timers.

- Configure 3 timer inputs ORed to the TI1 input channel by writing the TI1S bit in the TIMx_CR2 register to ‘1’.
- Program the time base: write the TIMx_ARR to the max value (the counter must be cleared by the TI1 change. Set the prescaler to get a maximum counter period longer than the time between 2 changes on the sensors.
- Program channel 1 in capture mode (TRC selected): write the CC1S bits in the TIMx_CCMR1 register to ‘11’. The user can also program the digital filter if needed.
- Program channel 2 in PWM 2 mode with the desired delay: write the OC2M bits to ‘111’ and the CC2S bits to ‘00’ in the TIMx_CCMR1 register.
- Select OC2REF as trigger output on TRGO: write the MMS bits in the TIMx_CR2 register to ‘101’.

In the advanced-control timer TIM1, the right ITR input must be selected as trigger input, the timer is programmed to generate PWM signals, the capture/compare control signals are preloaded (CCPC = 1 in the TIMx_CR2 register) and the COM event is controlled by the trigger input (CCUS = 1 in the TIMx_CR2 register). The PWM control bits (CCxE, OCxM) are written after a COM event for the next step (this can be done in an interrupt subroutine generated by the rising edge of OC2REF).

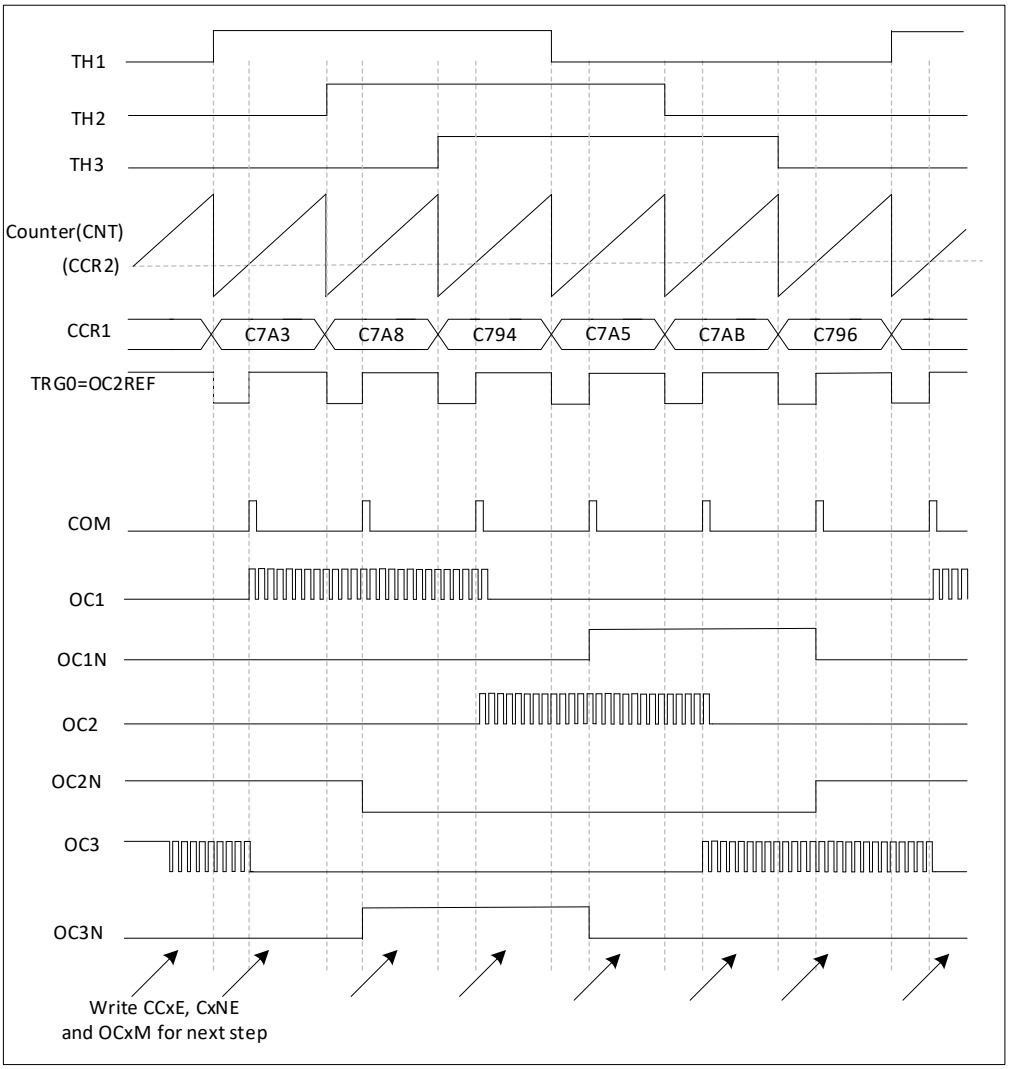

Figure 17-46 Example of Hall sensor interface

17.3.19. TIMx and external trigger synchronization

The TIMx timer can be synchronized with an external trigger in several modes: Reset mode, Gated mode and Trigger mode.

Slave mode: Reset mode

The counter and its prescaler can be reinitialized in response to an event on a trigger input.

Moreover, if the URS bit from the TIMx_CR1 register is low, an update event UEV is generated. Then all the preloaded registers (TIMx_ARR, TIMx_CCRx) are updated.

In the following example, the upcounter is cleared in response to a rising edge on TI1 input:

- Configure the channel 1 to detect rising edges on TI1. Configure the input filter duration (in this example, we don't need any filter, so we keep IC1F = 0000). The capture prescaler is not used for triggering, so there's no need to configure it. The CC1S bits select the input capture source only, CC1S = 01 in the TIMx_CCMR1 register. Write CC1P = 0 in TIMx_CCER register to validate the polarity (and detect rising edges only).
- Configure the timer in reset mode by writing SMS = 100 in TIMx_SMCR register. Select TI1 as the input source by writing TS = 101 in TIMx_SMCR register.
- Start the counter by writing CEN = 1 in the TIMx_CR1 register.

The counter starts counting on the internal clock, then behaves normally until TI1 rising edge. When TI1 rises, the counter is cleared and restarts from 0. In the meantime, the trigger flag is set (TIF bit in the TIMx_SR register) and an interrupt request, or a DMA request can be sent if enabled (depending on the TIE and TDE bits in TIMx_DIER register).

The following figure shows this behavior when the auto-reload register TIMx_ARR = 0x36. The delay between the rising edge on TI1 and the actual reset of the counter is due to the resynchronization circuit on TI1 input.

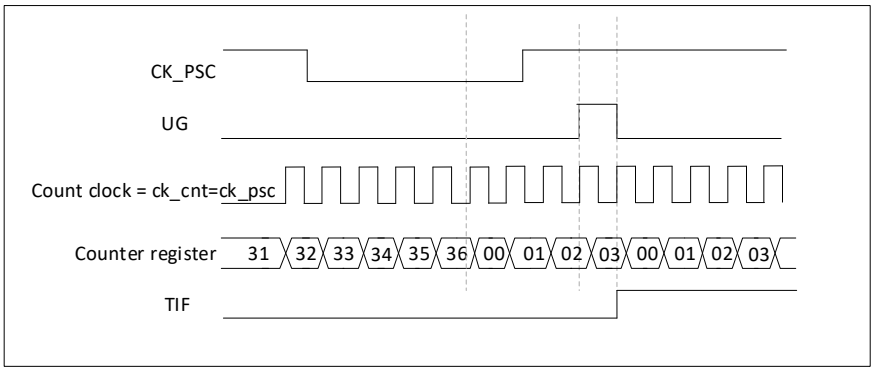

Figure 17-47 Control circuit in reset mode

Slave mode: Gated mode

The counter can be enabled depending on the level of a selected input.

In the following example, the upcounter counts only when TI1 input is low:

- Configure the channel 1 to detect low levels on TI1. Configure the input filter duration (in this example, we don't need any filter, so we keep IC1F = 0000). The capture prescaler is not used for triggering, so the user does not need to configure it. The CC1S bits select the input capture source only, CC1S = 01 in TIMx_CCMR1 register. Write CC1P = 1 in TIMx_CCER register to validate the polarity (and detect low level only).
- Configure the timer in gated mode by writing SMS = 101 in TIMx_SMCR register. Select TI1 as the input source by writing TS = 101 in TIMx_SMCR register.
- Enable the counter by writing CEN = 1 in the TIMx_CR1 register (in gated mode, the counter doesn't start if CEN = 0, whatever is the trigger input level).

The counter starts counting on the internal clock as long as TI1 is low and stops as soon as TI1 becomes high. The TIF flag in the TIMx_SR register is set both when the counter start or stops.

The delay between the rising edge on TI1 and the actual stop of the counter is due to the resynchronization circuit on TI1 input.

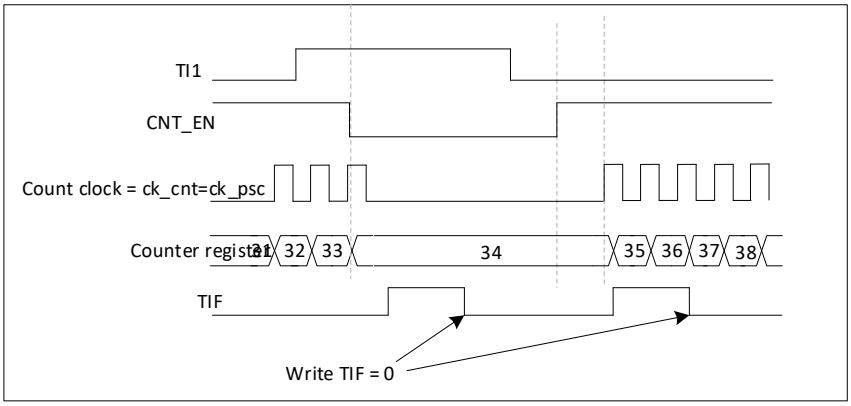

Figure 17-48 Control circuit in gated mode

The counter can start in response to an event on a selected input.

In the following example, the upcounter starts in response to a rising edge on TI2 input:

- Configure the channel 2 to detect rising edges on TI2. Configure the input filter duration (in this example, we don't need any filter, so we keep IC2F = 0000). The capture prescaler is not used for triggering, so there's no need to configure it. The CC2S bits are configured to select the input capture source only, CC2S = 01 in TIMx_CCMR1 register. Write CC2P = 1 in TIMx_CCER register to validate the polarity (and detect low level only).
- Configure the timer in trigger mode by writing SMS = 110 in TIMx_SMCR register. Select TI2 as the input source by writing TS = 110 in TIMx_SMCR register.

When a rising edge occurs on TI2, the counter starts counting on the internal clock and the TIF flag is set.

The delay between the rising edge on TI2 and the actual start of the counter is due to the resynchronization circuit on TI2 input.

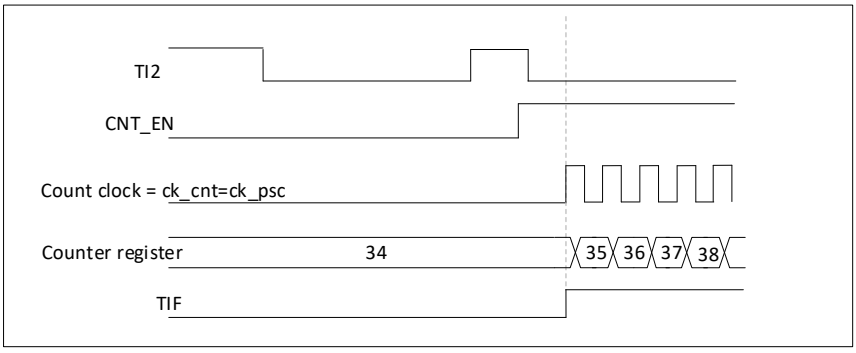

Figure 17-49 Control circuit in trigger mode

Slave mode: external clock mode 2 + trigger mode

The external clock mode 2 can be used in addition to another slave mode (except external clock mode 1 and encoder mode). In this case, the ETR signal is used as external clock input, and another input can be selected as trigger input (in reset mode, gated mode or trigger mode). It is recommended not to select ETR as TRGI through the TS bits of TIMx_SMCR register.

In the following example, the upcounter is incremented at each rising edge of the ETR signal as soon as a rising edge of TI1 occurs:

1. Configure the external trigger input circuit by programming the TIMx_SMCR register as follows:
   – ETF = 0000: no filter.

– ETPS = 00: prescaler disabled.

– ETP = 0: detection of rising edges on ETR and ECE = 1 to enable the external clock mode 2.

2. Configure the channel 1 as follows, to detect rising edges on TI:

– IC1F = 0000: no filter.

– The capture prescaler is not used for triggering and does not need to be configured.

– CC1S = 01 in TIMx_CCMR1 register to select only the input capture source

– CC1P = 0 in TIMx_CCER register to validate the polarity (and detect rising edge only).

3. Configure the timer in trigger mode by writing SMS = 110 in TIMx_SMCR register. Select TI1 as the input source by writing TS = 101 in TIMx_SMCR register.

A rising edge on TI1 enables the counter and sets the TIF flag. The counter then counts on ETR rising edges. The delay between the rising edge of the ETR signal and the actual reset of the counter is due to the resynchronization circuit on ETRP input.

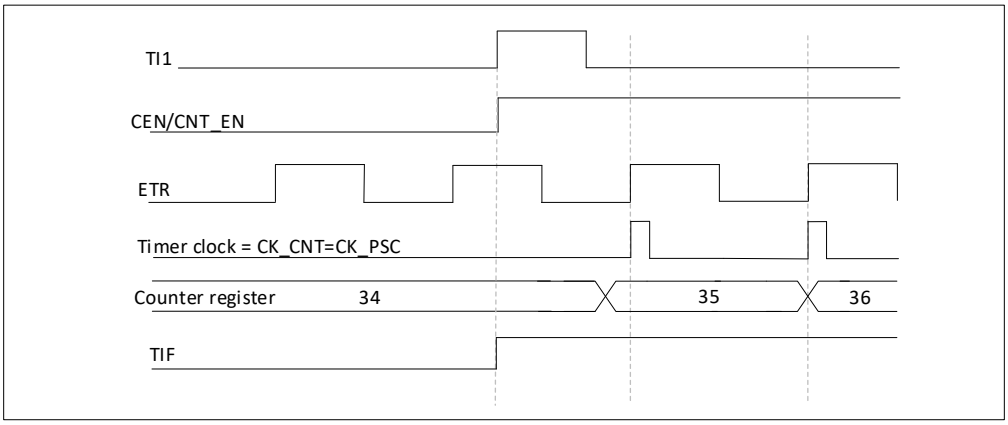

Figure 17-50 Control circuit in external clock mode 2 + trigger mode

17.3.20. Timer synchronization

The TIM timers are linked together internally for timer synchronization or chaining. When a timer is in master mode, it can reset, start, stop or clock the counter of another timer in slave mode.

17.3.21. Debug mode

When the chip enters the debug mode, according to the setting of DBG_TIMx_STOP in the DBG module, the TIMx counter can continue to work normally or stop working.

17.4. TIM1 registers

17.4.1. TIM1 control register1 (TIM1_CR1)

Address offset: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	CKD[1:0]		ARPE	CMS[1:0]		DIR	OPM	URS	UDIS	CEN
-	-	-	-	-	-	RW		RW	RW		RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved			
9:8	CKD[1:0]	RW	00	Clock division This bit-field indicates the division ratio between the timer clock (CK_INT) frequency and the dead-time and sampling clock (tDTS)used by the dead-time generators and the digital filters (ETR, TIx), 00: tDTS = tCK_INT 01: tDTS = 2 x tCK_INT 10: tDTS = 4 x tCK_INT 11: Reserved, do not program this value
7	ARPE	RW	0	Auto-reload preload enable 0: TIMx_ARR register is not buffered 1: TIMx_ARR register is buffered
6:5	CMS[1:0]	RW	00	Center-aligned mode selection 00: Edge-aligned mode. The counter counts up or down depending on the direction bit (DIR). 01: Center-aligned mode 1. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS = 00 in TIMx_CCMRx register) are set only when the counter is counting down. 10: Center-aligned mode 2. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS = 00 in TIMx_CCMRx register) are set only when the counter is counting up. 11: Center-aligned mode 3. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS = 00 in TIMx_CCMRx register) are set both when the counter is counting up or down. Note: It is not allowed to switch from edge-aligned mode to center-aligned mode as long as the counter is enabled (CEN = 1).
4	DIR	RW	0	Direction 0: Counter used as upcounter 1: Counter used as downcounter Note: This bit is read only when the timer is configured in Center-aligned mode or Encoder mode.
3	OPM	RW	0	One pulse mode 0: Counter is not stopped at update event 1: Counter stops counting at the next update event (clearing the bit CEN)
2	URS	RW	0	Update request source This bit is set and cleared by software to select the UEV event sources. 0: Any of the following events generate an update interrupt or DMA request if enabled. These events can be: – Counter overflow/underflow – Setting the UG bit – Update generation through the slave mode controller 1: Only counter overflow/underflow generates an update interrupt or DMA request if enabled.
1	UDIS	RW	0	Update disable This bit is set and cleared by software to enable/disable UEV event generation. 0: UEV enabled. The Update (UEV) event is generated by one of the following events: – Counter overflow/underflow – Setting the UG bit – Update generation through the slave mode controller Buffered registers are then loaded with their preload values. 1: UEV disabled. The Update event is not generated, shadow registers keep their value (ARR, PSC, CCRx). However the counter and the prescaler are reinitialized if the UG bit is set or if a hardware reset is received from the slave mode controller.

0	CEN	RW	0	Counter enable 0: Counter disabled 1: Counter enabled Note: External clock, gated mode and encoder mode can work only if the CEN bit has been previously set by software. However trigger mode can set the CEN bit automatically by hardware.

17.4.2. TIM1 control register2 (TIM1_CR2)

Address offset: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	OIS4	OIS3N	OIS3	OIS2N	OIS2	OIS1N	OIS1	TI1S	MMS[2:0]			CCDS	CCUS	Res	CCPC
-	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	-	RW

Bit	Name	R/W	Reset Value	Function
31:15	Reserved	-	0	Reserved, must be kept at reset value.
14	OIS4	RW		Output Idle state 4 (OC4 output) refer to OIS1 bit
13	OIS3N	RW	0	Output Idle state 3 (OC3N output) refer to OIS1N bit
12	OIS3	RW	0	Output Idle state 3 (OC3 output) refer to OIS1 bit
11	OIS2N	RW	0	Output Idle state 2 (OC2N output) refer to OIS1N bit
10	OIS2	RW	0	Output Idle state 2 (OC2 output) refer to OIS1 bit
9	OIS1N	RW	0	Output Idle state 1 (OC1N output) 0: OC1N = 0 after a dead-time when MOE = 0 1: OC1N = 1 after a dead-time when MOE = 0 Note: This bit cannot be modified as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in TIMx_BDTR register).
8	OIS1	RW	0	Output Idle state 1 (OC1 output) 0: OC1 = 0 (after a dead-time if OC1N is implemented) when MOE = 0 1: OC1 = 1 (after a dead-time if OC1N is implemented) when MOE = 0 Note: This bit cannot be modified as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in TIMx_BDTR register)
7	TI1S	RW	0	TI1 selection 0: The TIMx_CH1 pin is connected to TI1 input 1: The TIMx_CH1, CH2 and CH3 pins are connected to the TI1 input (XOR combination)
6:4	MMS[2:0]	RW	000	Master mode selection These bits allow to select the information to be sent in master mode to slave timers for synchronization (TRGO). The combination is as follows: 000: Reset - the UG bit from the TIMx_EGR register is used as trigger output (TRGO). If the reset is generated by the trigger input (slave mode controller configured in reset mode) then the signal on TRGO is delayed compared to the actual reset. 001: Enable - the Counter Enable signal CNT_EN is used as trigger output (TRGO). It is useful to start several timers at the same time or to control a window in which a slave timer is enable. The Counter Enable signal is generated by a logic OR between CEN control bit and the trigger input when configured in gated mode. When the Counter Enable signal is controlled by the trigger input, there is a delay on TRGO, except if the master/slave mode is selected (see the MSM bit description in TIMx_SMCR register).

				010: Update - The update event is selected as trigger output (TRGO). For instance a master timer can then be used as a prescaler for a slave timer. 011: Compare Pulse - The trigger output send a positive pulse when the CC1IF flag is to be set (even if it was already high), as soon as a capture or a compare match occurred(TRGO). 100: Compare - OC1REF signal is used as trigger output (TRGO) 101: Compare - OC2REF signal is used as trigger output (TRGO) 110: Compare - OC3REF signal is used as trigger output (TRGO) 111: Compare - OC4REF signal is used as trigger output (TRGO)
3	CCDS	RW	0	Capture/compare DMA selection 0: CCx DMA request sent when CCx event occurs 1: CCx DMA requests sent when update event occurs
2	CCUS	RW	0	Capture/compare control update selection 0: When capture/compare control bits are preloaded (CCPC = 1), they are updated by setting the COMG bit only 1: When capture/compare control bits are preloaded (CCPC = 1), they are updated by setting the COMG bit or when an rising edge occurs on TRGI *Note: This bit acts only on channels that have a complementary output.*
1	Res	-	0	Reserved, must be kept at reset value.
0	CCPC	RW	0	CCPC: Capture/compare preloaded control 0: CCxE, CCxNE and OCxM bits are not preloaded 1: CCxE, CCxNE and OCxM bits are preloaded, after having been written, they are updated only when a communication event (COM) occurs (COMG bit set or rising edge detected on TRGI, depending on the CCUS bit). Note: This bit acts only on channels that have a complementary output.

17.4.3. TIM1 slave mode control register (TIM1_SMCR)

Address offset: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ETP	ECE	ETPS[1:0]		ETF[3:0]				MSM	TS[2:0]			OCCS	SMS[2:0]		
RW	RW	RW		RW				RW	RW			RW	RW		

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			
15	ETP	RW	0	External trigger polarity This bit selects whether ETR or $\overline{ETR}$ is used for trigger operations 0: ETR is non-inverted, active at high level or rising edge. 1: ETR is inverted, active at low level or falling edge.
14	ECE	RW	0	External clock enable This bit enables External clock mode 2. 0: External clock mode 2 disabled 1: External clock mode 2 enabled. The counter is clocked by any active edge on the ETRF signal. Note: 1: Setting the ECE bit has the same effect as selecting external clock mode 1 with TRGI connected to ETRF (SMS = 111 and TS = 111). 2: It is possible to simultaneously use external clock mode 2 with the following slave modes: reset mode, gated mode and trigger

				mode. Nevertheless, TRGI must not be connected to ETRF in this case (TS bits must not be 111). 3: If external clock mode 1 and external clock mode 2 are enabled at the same time, the external clock input is ETRF.
13:12	ETPS[1:0]	RW	00	External trigger prescaler External trigger signal ETRP frequency must be at most 1/4 of TIMxCLK frequency. A prescaler can be enabled to reduce ETRP frequency. It is useful when inputting fast external clocks. 00: Prescaler OFF 01: ETRP frequency divided by 2 10: ETRP frequency divided by 4 11: ETRP frequency divided by 8
11:8	ETF[3:0]	RW	0000	External trigger filter This bit-field then defines the frequency used to sample ETRP signal and the length of the digital filter applied to ETRP. The digital filter is made of an event counter in which N consecutive events are needed to validate a transition on the output: 0000: No filter, sampling is done at fDTS 0001: fSAMPLING = fCK_INT, N = 2 0010: fSAMPLING = fCK_INT, N = 4 0011: fSAMPLING = fCK_INT, N = 8 0100: fSAMPLING = fDTS / 2, N = 6 0101: fSAMPLING = fDTS / 2, N = 8 0110: fSAMPLING = fDTS / 4, N = 6 0111: fSAMPLING = fDTS / 4, N = 8 1000: fSAMPLING = fDTS / 8, N = 6 1001: fSAMPLING = fDTS / 8, N = 8 1010: fSAMPLING = fDTS / 16, N = 5 1011: fSAMPLING = fDTS / 16, N = 6 1100: fSAMPLING = fDTS / 16, N = 8 1101: fSAMPLING = fDTS / 32, N = 5 1110: fSAMPLING = fDTS / 32, N = 6 1111: fSAMPLING = fDTS / 32, N = 8 Note: Care must be taken that fDTS is replaced in the formula by CK_INT when ETF[3:0] = 1, 2 or 3.
7	MSM	RW	0	Master/slave mode 0: No action 1: The effect of an event on the trigger input (TRGI) is delayed to allow a perfect synchronization between the current timer and its slaves (through TRGO). It is useful if we want to synchronize several timers on a single external event.
6:4	TS[2:0]	RW	000	Trigger selection This bit-field selects the trigger input to be used to synchronize the counter. 000: Internal Trigger 0 (ITR0) 001: Reserved 010: Internal Trigger 2 (ITR2) 011: Internal Trigger 3 (ITR3) 100: TI1 Edge Detector (TI1F_ED) 101: Filtered Timer Input 1 (TI1FP1) 110: Filtered Timer Input 2 (TI2FP2) 111: External Trigger input (ETRF) Note: These bits must be changed only when they are not used (e.g. when SMS = 000) to avoid wrong edge detections at the transition.
3	OCCS	RW	0	OCREF clear selection. This bit is used to select the OCREF clear source. 0:OCREF_CLR_INT is connected to the OCREF_CLR input 1: OCREF_CLR_INT is connected to ETRF
2:0	SMS[2:0]	RW	000	Slave mode selection When external signals are selected the active edge of the trigger signal (TRGI) is linked to the polarity selected on the external input (see Input Control register and Control Register description. 000: Slave mode disabled - if CEN = ‘1’ then the prescaler is clocked directly by the internal clock. 001: Encoder mode 1 - Counter counts up/down on TI2FP1 edge depending on TI1FP2 level. 010: Encoder mode 2 - Counter counts up/down on TI1FP2 edge depending on TI2FP1 level.

				011: Encoder mode 3 - Counter counts up/down on both TI1FP1 and TI2FP2 edges depending on the level of the other input. 100: Reset Mode - Rising edge of the selected trigger input (TRGI) reinitializes the counter and generates an update of the registers. 101: Gated Mode - The counter clock is enabled when the trigger input (TRGI) is high. The counter stops (but is not reset) as soon as the trigger becomes low. Both start and stop of the counter are controlled. 110: Trigger Mode - The counter starts at a rising edge of the trigger TRGI (but it is not reset). Only the start of the counter is controlled. 111: External Clock Mode 1 - Rising edges of the selected trigger (TRGI) clock the counter. Note: The gated mode must not be used if TI1F_ED is selected as the trigger input (TS = '100'). Indeed, TI1F_ED outputs 1 pulse for each transition on TI1F, whereas the gated mode checks the level of the trigger signal. Note: The clock of the slave timer must be enabled prior to receive events from the master timer, and must not be changed on-the-fly while triggers are received from the master timer.

Table 17-2 TIM1 Internal trigger connection

Slave TIM	ITR0(TS = 000)	ITR1(TS = 001)	ITR2(TS = 010)	ITR3(TS = 011)
TIM1	reserved	reserved	TIM3	TIM17 OC1

17.4.4. TIM1 DMA/interrupt enable register (TIM1_DIER)

Address offset: 0x0C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	**TDE**	**COMDE**	**CC4DE**	**CC3DE**	**CC2DE**	**CC1DE**	**UDE**	**BIE**	**TIE**	**COMIE**	**CC4IE**	**CC3IE**	**CC2IE**	**CC1IE**	**UIE**
-	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:15	Reserved			Reserved, must be kept at reset value.
14	TDE	RW	0	TDE: Trigger DMA request enable 0: Trigger DMA request disabled 1: Trigger DMA request enabled
13	COMDE	RW	0	COMDE: COM DMA request enable 0: COM DMA request disabled 1: COM DMA request enabled
12	CC4DE	RW	0	CC4DE: Capture/Compare 4 DMA request enable 0: CC4 DMA request disabled 1: CC4 DMA request enabled
11	CC3DE	RW	0	CC3DE: Capture/Compare 3 DMA request enable 0: CC3 DMA request disabled 1: CC3 DMA request enabled
10	CC2DE	RW	0	CC2DE: Capture/Compare 2 DMA request enable 0: CC2 DMA request disabled 1: CC2 DMA request enabled
9	CC1DE	RW	0	CC1DE: Capture/Compare 1 DMA request enable 0: CC1 DMA request disabled 1: CC1 DMA request enabled
8	UDE	RW	0	UDE: Update DMA request enable 0: Update DMA request disabled 1: Update DMA request enabled
7	BIE	RW	0	BIE: Break interrupt enable 0: Break interrupt disabled 1: Break interrupt enabled

6	TIE	RW	0	TIE: Trigger interrupt enable 0: Trigger interrupt disabled 1: Trigger interrupt enabled
5	COMIE	RW	0	COMIE: COM interrupt enable 0: COM interrupt disabled 1: COM interrupt enabled
4	CC4IE	RW	0	Capture/Compare 4 interrupt enable 0: CC4 interrupt disabled 1: CC4 interrupt enabled
3	CC3IE	RW	0	CC3IE: Capture/Compare 3 interrupt enable 0: CC3 interrupt disabled 1: CC3 interrupt enabled
2	CC2IE	RW	0	CC2IE: Capture/Compare 2 interrupt enable 0: CC2 interrupt disabled 1: CC2 interrupt enabled
1	CC1IE	RW	0	CC1IE: Capture/Compare 1 interrupt enable 0: CC1 interrupt disabled 1: CC1 interrupt enabled
0	UIE	RW	0	UIE: Update interrupt enable 0: Update interrupt disabled 1: Update interrupt enabled

17.4.5. TIM1 status register (TIM1_SR)

Address offset: 0x010

Reset value: 0x0000 0000

31	**30**	**29**	**28**	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	CC4OF	CC3OF	CC2OF	CC1OF	Res	BIF	TIF	COMIF	CC4IF	CC3IF	CC2IF	CC1F	UIF
-	-	-	Rc_w0	Rc_w0	Rc_w0	Rc_w0	-	Rc_w0	Rc_w0	Rc_w0	Rc_w0	Rc_w0	Rc_w0	Rc_w0	Rc_w0

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	0	Reserved, must be kept at reset value.
12	CC4OF	Rc_w0	0	Capture/Compare 4 overcapture flag refer to CC1OF description
11	CC3OF	Rc_w0	0	Capture/Compare 3 overcapture flag refer to CC1OF description
10	CC2OF	Rc_w0	0	Capture/Compare 2 overcapture flag refer to CC1OF description
9	CC1OF	Rc_w0	0	Capture/Compare 1 overcapture flag This flag is set by hardware only when the corresponding channel is configured in input capture mode. It is cleared by software by writing it to ‘0’. 0: No overcapture has been detected. 1: The counter value has been captured in TIMx_CCR1 register while CC1IF flag was already set
8	Res	Rc_w0	0	Reserved, must be kept at reset value.
7	BIF	Rc_w0	0	Break interrupt flag This flag is set by hardware as soon as the break input goes active. It can be cleared by software if the break input is not active. 0: No break event occurred. 1: An active level has been detected on the break input
6	TIF	Rc_w0	0	Trigger interrupt flag This flag is set by hardware on trigger event (active edge detected on TRGI input when the slave mode controller is enabled in all modes but gated mode.It is cleared by software. 0: No trigger event occurred. 1: Trigger interrupt pending
5	COMIF	Rc_w0	0	COM interrupt flag

				This flag is set by hardware on COM event (when Capture/compare Control bits - CCxE, CCxNE, OCxM - have been updated). It is cleared by software by writing it to '0'. 0: No COM event occurred. 1: COM interrupt pending.
4	CC4IF	Rc_w0	0	Capture/Compare 4 interrupt flag refer to CC1IF description
3	CC3IF	Rc_w0	0	Capture/Compare 3 interrupt flag refer to CC1IF description
2	CC2IF	Rc_w0	0	Capture/Compare 2 interrupt flag refer to CC1IF description
1	CC1IF	Rc_w0	0	Capture/Compare 1 interrupt flag **If channel CC1 is configured as output:** This flag is set by hardware when the counter matches the compare value, with some exception in center-aligned mode (refer to the CMS bits in the TIMx_CR1 register description). It is cleared by software. 0: No match. 1: The content of the counter TIMx_CNT matches the content of the TIMx_CCR1 register. When the contents of TIMx_CCR1 are greater than the contents of TIMx_ARR, the CC1IF bit goes high on the counter overflow (in upcounting and up/down-counting modes) or underflow (in downcounting mode) **If channel CC1 is configured as input:** This bit is set by hardware on a capture. It is cleared by software or by reading the TIMx_CCR1 register. 0: No input capture occurred 1: The counter value has been captured in TIMx_CCR1 register (An edge has been detected on IC1 which matches the selected polarity)
0	UIF	Rc_w0	0	Update interrupt flag This bit is set by hardware on an update event. It is cleared by software. 0: No update occurred. 1: Update interrupt pending. This bit is set by hardware when the registers are updated: –At overflow or underflow regarding the repetition counter value (update if repetition counter = 0) and if the UDIS = 0 in the TIMx_CR1 register. –When CNT is reinitialized by software using the UG bit in TIMx_EGR register, if URS = 0 and UDIS = 0 in the TIMx_CR1 register. –When CNT is reinitialized by a trigger event (refer to TIM1 slave mode control register (TIM1_SMCR)), if URS = 0 and UDIS = 0 in the TIMx_CR1 register.

17.4.6. TIM1 event generation register (TIM1_EGR)

Address offset: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	BG	TG	COMG	CC4G	CC3G	CC2G	CC1G	UG
-	-	-	--	-	-	-	-	W	W	W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	0	Reserved, must be kept at reset value.
7	BG	W	0	Break generation This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action

				1: A break event is generated. MOE bit is cleared and BIF flag is set. Related interrupt or DMA transfer can occur if enabled.
6	TG	W	0	Trigger generation This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: The TIF flag is set in TIMx_SR register. Related interrupt or DMA transfer can occur if enabled.
5	COMG	W	0	Capture/Compare control update generation This bit can be set by software, it is automatically cleared by hardware 0: No action 1: When CCPC bit is set, it allows to update CCxE, CCxNE and OCxM bits Note: This bit acts only on channels having a complementary output.
4	CC4G	W	0	CC4G: Capture/Compare 4 generation Refer to CC1G description
3	CC3G	W	0	CC3G: Capture/Compare 3 generation Refer to CC1G description
2	CC2G	W	0	CC2G: Capture/Compare 2 generation Refer to CC1G description
1	CC1G	W	0	Capture/Compare 1 generation This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: A capture/compare event is generated on channel 1: **If channel CC1 is configured as output:** CC1IF flag is set, Corresponding interrupt or DMA request is sent if enabled. **If channel CC1 is configured as input:** The current value of the counter is captured in TIMx_CCR1 register. The CC1IF flag is set, the corresponding interrupt or DMA request is sent if enabled. The CC1OF flag is set if the CC1IF flag was already high.
0	UG	W	0	Update generation This bit can be set by software, it is automatically cleared by hardware. 0: No action 1: Reinitialize the counter and generates an update of the registers. Note that the prescaler counter is cleared too (anyway the prescaler ratio is not affected). The counter is cleared if the center-aligned mode is selected or if DIR = 0 (upcounting), else it takes the auto-reload value (TIMx_ARR) if DIR = 1 (downcounting).

17.4.7. TIM1 capture/compare mode register1 (TIM1_CCMR1)

Address offset: 0x18

Reset value: 0x0000 0000

Output compare mode:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
OC2CE	OC2M[2:0]			OC2PE	CO2FE	CC2S[1:0]		OC1CE	OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-		Reserved, must be kept at reset value.
15	OC2CE	RW	0	Output Compare 2 clear enable
14:12	OC2M[2:0]	RW	000	Output Compare 2 mode

11	OC2PE	RW	0	Output Compare 2 preload enable
10	OC2FE	RW	0	Output Compare 2 fast enable
9:8	CC2S[1:0]	RW	00	Capture/Compare 2 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC2 channel is configured as output 01: CC2 channel is configured as input, IC2 is mapped on TI2 10: CC2 channel is configured as input, IC2 is mapped on TI1 11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode is working only if an internal trigger input is selected through the TS bit (TIMx_SMCR register) Note: CC2S bits are writable only when the channel is OFF (CC2E = '0' in TIMx_CCER).
7	OC1CE	RW	0	Output Compare 1 clear enable OC1CE: Output Compare 1 Clear Enable 0: OC1Ref is not affected by the ETRF Input 1: OC1Ref is cleared as soon as a High level is detected on ETRF input
6:4	OC1M[2:0]	RW	00	Output Compare 1 mode These bits define the behavior of the output reference signal OC1REF from which OC1 and OC1N are derived. OC1REF is active high whereas OC1 and OC1N active level depends on CC1P and CC1NP bits. 000: Frozen - The comparison between the output compare register TIMx_CCR1 and the counter TIMx_CNT has no effect on the outputs (this mode is used to generate a timing base). 001: Set channel 1 to active level on match. OC1REF signal is forced high when the counter TIMx_CNT matches the capture/compare register 1 (TIMx_CCR1). 010: Set channel 1 to inactive level on match. OC1REF signal is forced low when the counter TIMx_CNT matches the capture/compare register 1 (TIMx_CCR1). 011: Toggle - OC1REF toggles when TIMx_CNT = TIMx_CCR1. 100: Force inactive level - OC1REF is forced low. 101: Force active level - OC1REF is forced high. 110: PWM mode 1 - In upcounting, channel 1 is active as long as TIMx_CNT < TIMx_CCR1 else inactive. In downcounting, channel 1 is inactive (OC1REF = '0') as long as TIMx_CNT > TIMx_CCR1 else active (OC1REF = '1'). 111: PWM mode 2 - In upcounting, channel 1 is inactive as long as TIMx_CNT < TIMx_CCR1 else active. In downcounting, channel 1 is active as long as TIMx_CNT > TIMx_CCR1 else inactive. Note: 1: These bits can not be modified as long as LOCK level 3 has been programmed(LOCK bits in TIMx_BDTR register) and CC1S = '00' (the channel is configured in output). 2: In PWM mode 1 or 2, the OCREF level changes only when the result of the comparison changes or when the output compare mode switches from "frozen" mode to "PWM" mode. 3: On channels having a complementary output, this bit field is preloaded. If the CCPC bit is set in the TIMx_CR2 register then the OC1M active bits take the new value from the preloaded bits only when a COM event is generated.
3	OC1PE	RW	0	Output Compare 1 preload enable 0: Preload register on TIMx_CCR1 disabled. TIMx_CCR1 can be written at anytime, the new value is taken in account immediately. 1: Preload register on TIMx_CCR1 enabled. Read/Write operations access the preload register. TIMx_CCR1 preload value is loaded in the active register at each update event.

				Note: 1: These bits can not be modified as long as LOCK level 3 has been programmed (LOCK bits in TIMx_BDTR register) and CC1S = ’00’ (the channel is configured in output). 2: The PWM mode can be used without validating the pre-load register only in one pulse mode (OPM bit set in TIMx_CR1 register). Else the behavior is not guaranteed.
2	OC1FE	RW	0	Output Compare 1 fast enable This bit is used to accelerate the effect of an event on the trigger in input on the CC output. 0: CC1 behaves normally depending on counter and CCR1 values even when the trigger is ON. The minimum delay to activate CC1 output when an edge occurs on the trigger input is 5 clock cycles. 1: An active edge on the trigger input acts like a compare match on CC1 output. Then, OC is set to the compare level independently from the result of the comparison. Delay to sample the trigger input and to activate CC1 output is reduced to 3 clock cycles. OCFE acts only if the channel is configured in PWM1 or PWM2 mode.
1:0	CC1S[1:0]	RW	00	Capture/Compare 1 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC1 channel is configured as output 01: CC1 channel is configured as input, IC1 is mapped on TI1 10: CC1 channel is configured as input, IC1 is mapped on TI2 11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIMx_SMCR register) Note: CC1S bits are writable only when the channel is OFF (CC1E = ‘0’ in TIMx_CCER).

Input Capture mode:

31	**30**	**29**	**28**	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
IC2F[3:0]				IC2PSC[1:0]		CC2S[1:0]		IC1F[3:0]				IC1PSC[1:0]		CC1S[1:0]	
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-		Reserved, must be kept at reset value.
15:12	IF2F	RW	0000	Input capture 2 filter
11:10	IC2PSC[1:0]	RW	00	Input capture 2 prescaler
9:8	CC2S[1:0]	RW	0	Capture/Compare 2 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC2 channel is configured as output 01: CC2 channel is configured as input, IC2 is mapped on TI2 10: CC2 channel is configured as input, IC2 is mapped on TI1 11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIMx_SMCR register) Note: CC2S bits are writable only when the channel is OFF (CC2E = ‘0’ in TIMx_CCER)
7:4	IC1F[3:0]	RW	0000	Input capture 1 filter This bit-field defines the frequency used to sample TI1 input and the length of the digital filter applied to TI1. The digital filter is made of an event counter in which N consecutive events are needed to validate a transition on the output: 0000: No filter, sampling is done at fDTS 0001: fSAMPLING = fCK_INT, N = 2 0010: fSAMPLING = fCK_INT, N = 4

				0011: fSAMPLING = fCK_INT, N = 8 0100: fSAMPLING = fDTS / 2, N = 6 0101: fSAMPLING = fDTS / 2, N = 8 0110: fSAMPLING = fDTS / 4, N = 6 0111: fSAMPLING = fDTS / 4, N = 8 1000: fSAMPLING = fDTS / 8, N = 6 1001: fSAMPLING = fDTS / 8, N = 8 1010: fSAMPLING = fDTS / 16, N = 5 1011: fSAMPLING = fDTS / 16, N = 6 1100: fSAMPLING = fDTS / 16, N = 8 1101: fSAMPLING = fDTS / 32, N = 5 1110: fSAMPLING = fDTS / 32, N = 6 1111: fSAMPLING = fDTS / 32, N = 8 Note: Care must be taken that fDTS is replaced in the formula by CK_INT when ICxF[3:0] = 1, 2 or 3.
3:2	IC1PSC[1:0]	RW	00	Input capture 1 prescaler This bit-field defines the ratio of the prescaler acting on CC1 input (IC1). The prescaler is reset as soon as CC1E = ’0’ (TIMx_CCER register). 00: no prescaler, capture is done each time an edge is detected on the capture input 01: capture is done once every 2 events 10: capture is done once every 4 events 11: capture is done once every 8 events
1:0	CC1S[1:0]	RW	00	Capture/Compare 1 Selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC1 channel is configured as output 01: CC1 channel is configured as input, IC1 is mapped on TI1 10: CC1 channel is configured as input, IC1 is mapped on TI2 11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIMx_SMCR register) Note: CC1S bits are writable only when the channel is OFF (CC1E = ‘0’ in TIMx_CCER).

17.4.8. TIM1 capture/compare mode register 2 (TIM1_CCMR2)

Address offset: 0x1C

Reset value: 0x0000 0000

Output compare mode:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
OC4CE	OC4M[2:0]			OC4PE	CO4FE	CC4S[1:0]		OC3CE	OC3M[2:0]			OC3PE	OC3FE	CC3S[1:0]	
IC4F[3:0]				IC4PSC[1:0]				IC3F[3:0]				IC3PSC[1:0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-		Reserved, must be kept at reset value.
15	OC4CE	RW	0	Output compare 4 clear enable
14:12	OC4M[2:0]	RW	000	Output compare 4 mode
11	OC4PE	RW	0	Output compare 4 preload enable
10	OC4FE	RW	0	Output compare 4 fast enable
9:8	CC4S[1:0]	RW	00	Capture/Compare 4 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC4 channel is configured as output

				01: CC4 channel is configured as input, IC4 is mapped on TI4 10: CC4 channel is configured as input, IC4 is mapped on TI3 11: CC4 channel is configured as input, IC4 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIMx_SMCR register) Note: CC4S bits are writable only when the channel is OFF (CC4E = '0' in TIMx_CCER).
7	OC3CE	RW	0	Output compare 3 clear enable
6:4	OC3M[2:0]	RW	00	Output compare 3 mode
3	OC3PE	RW	0	Output compare 3 preload enable
2	OC3FE	RW	0	Output compare 3 fast enable
1:0	CC3S[1:0]	RW	00	Capture/Compare 3 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC3 channel is configured as output 01: CC3 channel is configured as input, IC3 is mapped on TI3 10: CC3 channel is configured as input, IC3 is mapped on TI4 11: CC3 channel is configured as input, IC3 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIMx_SMCR register) Note: CC3S bits are writable only when the channel is OFF (CC3E = '0' in TIMx_CCER)

Input Capture mode:

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-		Reserved, must be kept at reset value.
15:12	IC4F	RW	0000	Input capture 4 filter
11:10	IC4PSC	RW	00	Input capture 4 prescaler
9:8	CC4S	RW	00	Capture/Compare 4 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC4 channel is configured as output 01: CC4 channel is configured as input, IC4 is mapped on TI4 10: CC4 channel is configured as input, IC4 is mapped on TI3 11: CC4 channel is configured as input, IC4 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIMx_SMCR register) Note: CC4S bits are writable only when the channel is OFF (CC4E = '0' in TIMx_CCER)
7:4	IC3F	RW	0000	Input capture 3 filter
3:2	IC3PSC	RW	00	Input capture 3 prescaler
1:0	OC3S	RW	00	Capture/compare 3 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC3 channel is configured as output 01: CC3 channel is configured as input, IC3 is mapped on TI3 10: CC3 channel is configured as input, IC3 is mapped on TI4 11: CC3 channel is configured as input, IC3 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIMx_SMCR register) Note: CC3S bits are writable only when the channel is OFF (CC3E = '0' in TIMx_CCER).

17.4.9. TIM1 capture/compare enable register (TIM1_CCER)

Address offset: 0x20

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	CC4P	CC4E	CC3NP	CC3NE	CC3P	CC3E	CC2NP	CC2NE	CC2P	CC2E	CC1NP	CC1NE	CC1P	CC1E
-	-	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:14	Reserved	-	0	Reserved, must be kept at reset value.
13	CC4P	RW	0	Capture/Compare 4 output polarity refer to CC1P description
12	CC4E	RW	0	Capture/Compare 4 output enable refer to CC1E description
11	CC3NP	RW	0	Capture/Compare 3 complementary output polarity refer to CC1NP description
10	CC3NE	RW	0	Capture/Compare 3 complementary output enable refer to CC1NE description
9	CC3P	RW	0	Capture/Compare 3 output polarity refer to CC1P description
8	CC3E	RW	0	Capture/Compare 3 output enable refer to CC1E description
7	CC2NP	RW	0	Capture/Compare 2 complementary output polarity refer to CC1NP description
6	CC2NE	RW	0	Capture/Compare 2 complementary output enable refer to CC1NE description
5	CC2P	RW	0	Capture/Compare 2 output polarity refer to CC1P description
4	CC2E	RW	0	Capture/Compare 2 output enable refer to CC1E description
3	CC1NP	RW	0	Capture/Compare 1 complementary output polarity **CC1 channel configuration as output:** 0: OC1N active high. 1: OC1N active low. **CC1 channel configuration as input:** This bit is used in conjunction with CC1P to define the polarity of TI1FP1 and TI2FP1. Refer to CC1P description. Note: On channels having a complementary output, this bit is preloaded. If the CCPC bit is set in the TIMx_CR2 register then the CC1NP active bit takes the new value from the preloaded bits only when a Commutation event is generated. Note: This bit is not writable as soon as LOCK level 2 or 3 has been programmed (LOCK bits in TIMx_BDTR register) and CC1S = ”00” (the channel is configured in output).
2	CC1NE	RW	0	Capture/Compare 1 complementary output enable 0: Off - OC1N is not active. OC1N level is then function of MOE, OSSI, OSSR, OIS1, OIS1N and CC1E bits. 1: On - OC1N signal is output on the corresponding output pin depending on MOE, OSSI, OSSR, OIS1, OIS1N and CC1E bits. Note: On channels having a complementary output, this bit is preloaded. If the CCPC bit is set in the TIMx_CR2 register then the CC1NE active bit takes the new value from the preloaded bits only when a Commutation event is generated.
1	CC1P	RW	0	Capture/Compare 1 output polarity **CC1 channel configured as output:** 0: OC1 active high 1: OC1 active low **CC1 channel configured as input:** CC1NP/CC1P bits select the active polarity of TI1FP1 and TI2FP1 for trigger or capture operations. 00: non-inverted/rising edge The circuit is sensitive to TIxFP1 rising edge (capture or trigger operations in reset, external clock or trigger mode),

				TIxFP1 is not inverted (trigger operation in gated mode or encoder mode). 01: inverted/falling edge The circuit is sensitive to TIxFP1 falling edge (capture or trigger operations in reset, external clock or trigger mode), TIxFP1 is inverted (trigger operation in gated mode or encoder mode). 10: reserved, do not use this configuration. 11: non-inverted/both edges The circuit is sensitive to both TIxFP1 rising and falling edges (capture or trigger operations in reset, external clock or trigger mode), TIxFP1 is not inverted (trigger operation in gated mode). This configuration must not be used in encoder mode. Note: On channels having a complementary output, this bit is preloaded. If the CCPC bit is set in the TIMx_CR2 register then the CC1P active bit takes the new value from the preloaded bits only when a Commutation event is generated. Note: This bit is not writable as soon as LOCK level 2 or 3 has been programmed (LOCK bits in TIMx_BDTR register).
0	CC1E	RW	0	Capture/Compare 1 output enable CC1 channel configured as output: 0: Off - OC1 is not active. OC1 level is then function of MOE, OSSI, OSSR, OIS1, OIS1N and CC1NE bits. 1: On - OC1 signal is output on the corresponding output pin depending on MOE, OSSI, OSSR, OIS1, OIS1N and CC1NE bits. CC1 channel configured as input: This bit determines if a capture of the counter value can actually be done into the input capture/compare register 1 (TIMx_CCR1) or not. 0: Capture disabled. 1: Capture enabled. Note: On channels having a complementary output, this bit is preloaded. If the CCPC bit is set in the TIMx_CR2 register then the CC1E active bit takes the new value from the preloaded bits only when a Commutation event is generated.

Table 17-3 Output control bits for complementary OCx and OCxN channels with break feature

Control bits					Output states(1)	
MOE bit	**OSSI bit**	**OSSR bit**	**CCxE bit**	**CCxNE bit**	**OCx output state**	**OCxN output state**
1	x	0	0	1	Output Disabled (not driven by the timer), OCx = 0, OCx_EN = 0	Output Disabled (not driven by the timer), OCxN = 0, OCxN_EN = 0
		0	1	0	Output Disabled (not driven by the timer), OCx = 0, OCx_EN = 0	OCxREF + Polarity OCxN = OCxREF xor CCxNP, OCxN_EN = 1
		0	1	1	OCxREF + Polarity OCx = OCxREF xor CCxP, OCx_EN = 1	Output Disabled (not driven by the timer) OCxN = 0, OCxN_EN = 0
		1	0	0	OCREF + Polarity + dead-time OCx_EN = 1	Complementary to OCREF (not OCREF) + Polarity + dead-time OCxN_EN = 1
		1	0	1	Output Disabled (not driven by the timer) OCx = CCxP, OCx_EN = 0	Output Disabled (not driven by the timer) OCxN = CCxNP, OCxN_EN = 0
		1	1	0	Off-State (output enabled with inactive state) OCx = CCxP, OCx_EN = 1	OCxREF + Polarity OCxN = OCxREF xor CCxNP, OCxN_EN = 1
		1	1	1	OCxREF + Polarity OCx = OCxREF xor CCxP, OCx_EN = 1	Off-State (output enabled with inactive state) OCxN = CCxNP, OCxN_EN = 1
		0	0	0	OCREF + Polarity + dead-time OCx_EN = 1	Complementary to OCREF (not OCREF) + Polarity + dead-time OCxN_EN = 1
0	0	X	0	0	Output Disabled (not driven by the timer)	

	0		0	1	Asynchronously: OCx = CCxP, OCx_EN = 0, OCxN = CCxNP, OCxN_EN = 0 Then if the clock is present: OCx = OISx and OCxN = OISxN after a dead-time, assuming that OISx and OISxN do not correspond to OCX and OCxN both in active state.
	0		1	0	
	0		1	1	
	1		0	0	
	1		0	1	Off-State (output enabled with inactive state) Asynchronously: OCx = CCxP, OCx_EN = 1, OCxN = CCxNP, OCxN_EN = 1 Then if the clock is present: OCx = OISx and OCxN = OISxN after a dead-time, assuming that OISx and OISxN do not correspond to OCX and OCxN both in active state
	1		1	0	
	1		1	1	

1. When both outputs of a channel are not used (CCxE = CCxNE = 0), the OISx, OISxN, CCxP and CCxNP bits must be kept cleared.

Note: The state of the external I/O pins connected to the complementary OCx and OCxN channels depends on the OCx and OCxN channel state and the GPIOand AFIO registers.

17.4.10. TIM1 counter (TIM1_CNT)

Address offset: 0x24

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CNT[15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			Reserved, must be kept at reset value.
15:0	CNT[15:0]	RW	0	Counter value

17.4.11. TIM1 prescaler (TIM1_PSC)

Address offset: 0x28

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PSC[15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			Reserved, must be kept at reset value.
15:0	PSC[15:0]	RW	0	Prescaler value The counter clock frequency (CK_CNT) is equal to fCK_PSC / (PSC[15:0] + 1). PSC contains the value to be loaded in the active pre-scaler register at each update event (including when the counter is cleared through UG bit of TIMx_EGR register or through trigger controller when configured in “reset mode”).

17.4.12. TIM1 auto-reload register (TIM1_ARR)

Address offset: 0x2c

Reset value: 0x0000 FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ARR[15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			Reserved, must be kept at reset value.
15:0	ARR[15:0]	RW	0	Auto-reload value ARR is the value to be loaded in the actual auto-reload register. The counter is blocked while the auto-reload value is null.

17.4.13. TIM1 repetition counter register (TIM1_RCR)

Address offset: 0x30

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	**Res**	**Res**	**Res**	**Res**	**Res**	**Res**	**Res**	REP[7:0]							
-	-	-	-	-	-	-	-	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:8	Reserved			Reserved, must be kept at reset value.
7:0	REP[7:0]	RW	0	Repetition counter value These bits allow the user to set-up the update rate of the compare registers (i.e. periodic transfers from preload to active registers) when preload registers are enable, as well as the update interrupt generation rate, if this interrupt is enable. Each time the REP_CNT related downcounter reaches zero, an update event is generated and it restarts counting from REP value. As REP_CNT is reloaded with REP value only at the repetition update event U_RC, any write to the TIMx_RCR register is not taken in account until the next repetition update event. It means in PWM mode (REP+1) corresponds to: – the number of PWM periods in edge-aligned mode – the number of half PWM period in center-aligned mode.

17.4.14. TIM1 capture/compare register 1 (TIM1_CCR1)

Address offset: 0x34

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR1[15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			Reserved, must be kept at reset value.
15:0	CCR1[15:0]	RW	0	Capture/Compare 1 value **If channel CC1 is configured as output:** CCR1 is the value to be loaded in the actual capture/compare 1 register (preload value). It is loaded permanently if the preload feature is not selected in the TIMx_CCMR1 register (bit OC1PE). Else the preload value is copied in the active capture/compare 1 register when an

				update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC1 output. **If channel CC1 is configured as input:** CCR1 is the counter value transferred by the last input capture 1 event (IC1).

17.4.15. TIM1 capture/compare register 2 (TIM1_CCR2)

Address offset: 0x38

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR2[15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			Reserved, must be kept at reset value.
15:0	CCR2[15:0]	RW	0	Capture/Compare 2 value **If channel CC2 is configured as output:** CCR2 is the value to be loaded in the actual capture/compare 2 register (preload value). It is loaded permanently if the preload feature is not selected in the TIMx_CCMR2 register (bit OC2PE). Else the preload value is copied in the active capture/compare 2 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signalled on OC2 output. **If channel CC2 is configured as input:** CCR2 is the counter value transferred by the last input capture 2 event (IC2).

17.4.16. TIM1 capture/compare register 3 (TIM1_CCR3)

Address offset: 0x3C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR3[15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			Reserved, must be kept at reset value.
15:0	CCR3[15:0]	RW	0	Capture/Compare value **If channel CC3 is configured as output:** CCR3 is the value to be loaded in the actual capture/compare 3 register (preload value). It is loaded permanently if the preload feature is not selected in the TIMx_CCMR3 register (bit OC3PE). Else the preload value is copied in the active capture/compare 3 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signalled on OC3 output. **If channel CC3 is configured as input:** CCR3 is the counter value transferred by the last input capture 3 event (IC3).

17.4.17. TIM1 capture/compare register 4 (TIM1_CCR4)

Address offset: 0x40

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR4[15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			Reserved, must be kept at reset value.
15:0	CCR4[15:0]	RW	0	Capture/Compare value **If channel CC4 is configured as output:** CCR4 is the value to be loaded in the actual capture/compare 4 register (preload value). It is loaded permanently if the preload feature is not selected in the TIMx_CCMR4 register (bit OC4PE). Else the preload value is copied in the active capture/compare 4 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signalled on OC4 output. **If channel CC4 is configured as input:** CCR4 is the counter value transferred by the last input capture 4 event (IC4).

17.4.18. TIM1 break and dead-time register (TIM1_BDTR)

Address offset: 0x44

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MOE	AOE	BKP	BKE	OSSR	OSSI	LOCK[1:0]		DTG[7:0]							
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	RW	0	Reserved, must be kept at reset value.
15	MOE	RW	0	Main output enable This bit is cleared asynchronously by hardware as soon as the break input is active. It is set by software or automatically depending on the AOE bit. It is acting only on the channels which are configured in output. 0: OC and OCN outputs are disabled or forced to idle state. 1: OC and OCN outputs are enabled if their respective enable bits are set (CCxE, CCxNE in TIMx_CCER register).
14	AOE	RW	0	Automatic output enable 0: MOE can be set only by software 1: MOE can be set by software or automatically at the next update event (if the break input is not be active) Note: This bit cannot be modified as long as LOCK level 1 has been programmed (LOCK bits in TIMx_BDTR register).
13	BKP	RW	0	Break polarity 0: Break input BRK is active low 1: Break input BRK is active high Note: This bit can not be modified as long as LOCK level 1 has been programmed (LOCK bits in TIMx_BDTR register). Note: Any write operation to this bit takes a delay of 1 APB clock cycle to become effective.
12	BKE	RW	0	Break enable

				0: Break inputs (BRK and CCS clock failure event) disabled 1, Break inputs (BRK and CCS clock failure event) enabled Note: This bit cannot be modified when LOCK level 1 has been programmed (LOCK bits in TIMx_BDTR register). Note: Any write operation to this bit takes a delay of 1 APB clock cycle to become effective.
11	OSSR	RW	0	Off-state selection for Run mode This bit is used when MOE = 1 on channels having a complementary output which are configured as outputs. OSSR is not implemented if no complementary output is implemented in the timer. 0: When inactive, OC/OCN outputs are disabled (OC/OCN enable output signal = 0). 1: When inactive, OC/OCN outputs are enabled with their inactive level as soon as CCxE = 1 or CCxNE = 1. Then, OC/OCN enable output signal = 1 Note: This bit can not be modified as soon as the LOCK level 2 has been programmed (LOCK bits in TIMx_BDTR register).
10	OSSI	RW	0	Off-state selection for Idle mode This bit is used when MOE = 0 on channels configured as outputs. 0: When inactive, OC/OCN outputs are disabled (OC/OCN enable output signal = 0). 1: When inactive, OC/OCN outputs are forced first with their idle level as soon as CCxE = 1 or CCxNE = 1. OC/OCN enable output signal = 1) Note: This bit can not be modified as soon as the LOCK level 2 has been programmed (LOCK bits in TIMx_BDTR register).
9:8	LOCK[1:0]	RW	00	Lock configuration These bits offer a write protection against software errors. 00: LOCK OFF - No bit is write protected. 01: LOCK Level 1 = DTG bits in TIMx_BDTR register, OISx and OISxN bits in TIMx_CR2 register and BKE/BKP/AOE bits in TIMx_BDTR register can no longer be written. 10: LOCK Level 2 = LOCK Level 1 + CC Polarity bits (CCxP/CCxNP bits in TIMx_CCER register, as long as the related channel is configured in output through the CCxS bits) as well as OSSR and OSSI bits can no longer be written. 11: LOCK Level 3 = LOCK Level 2 + CC Control bits (OCxM and OCxPE bits in TIMx_CCMRx registers, as long as the related channel is configured in output through the CCxS bits) can no longer be written. Note: The LOCK bits can be written only once after the reset. Once the TIMx_BDTR register has been written, their content is frozen until the next reset.
7:0	DTG[7:0]	RW	0000 0000	Dead-time generator setup This bit-field defines the duration of the dead-time inserted between the complementary outputs. DT correspond to this duration. DTG[7:5] = 0xx = > DT = DTG[7:0]x tdtg with tdtg = tDTS. DTG[7:5] = 10x = > DT = (64+DTG[5:0])xtdtg with Tdtg = 2xtDTS. DTG[7:5] = 110 = > DT = (32+DTG[4:0])xtdtg with Tdtg = 8xtDTS. DTG[7:5] = 111 = > DT = (32+DTG[4:0])xtdtg with Tdtg = 16xtDTS. Example if TDTS = 125 ns (8 MHz), dead-time possible values are: 0 to 15875 ns by 125 ns steps, 16 us to 31750 ns by 250 ns steps, 32 us to 63 us by 1 us steps, 64 us to 126 us by 2 us steps

				Note: This bit-field can not be modified as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in TIMx_BDTR register).

17.4.19. TIM1 DMA control register (TIM1_DCR)

Address offset: 0x48

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	DBL[4:0]					Res			DBA[4:0]				
-	-	-	RW	RW	RW	RW	RW	-	-	-	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:13	Reserved			Reserved, must be kept at reset value.
12:8	DBL[4:0]	RW	0 0000	DMA burst length This 5-bit vector defines the number of DMA transfers (the timer recognizes a burst transfer when a read or a write access is done to the TIMx_DMAR address) 00000: 1 transfer 00001: 2 transfers 00010: 3 transfers ... 10001: 18 transfers
7:5	Reserved	RW	0	Reserved, must be kept at reset value.
4:0	DBA[4:0]	RW	0 0000	DMA base address This 5-bit vector defines the base-address for DMA transfers (when read/write access are done through the TIMx_DMAR address). DBA is defined as an offset starting from the address of the TIMx_CR1 register. Example: 00000: TIMx_CR1, 00001: TIMx_CR2, 00010: TIMx_SMCR, ... **Example:** Let us consider the following transfer: DBL = 7 transfers and DBA = TIMx_CR1. In this case the transfer is done to/from 7 registers starting from the TIMx_CR1 address.

17.4.20. TIM1 DMA address for full transfer (TIM1_DMAR)

Address offset: 0x4C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DMAB[15:0]															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			Reserved, must be kept at reset value.
15:0	DMAB[31:0]	RW	0	DMAB[15:0]: DMA register for burst accesses A read or write operation to the DMAR register accesses the register located at the address (TIMx_CR1 address) + (DBA + DMA index) x 4 where TIMx_CR1 address is the address of the control register 1, DBA is the DMA base address configured in TIMx_DCR register, DMA index is automatically controlled by the DMA transfer, and ranges from 0 to DBL (DBL configured in TIMx_DCR).

17.4.21. TIM1 register map

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x000	TIM1_CR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CKD[1:0]		ARPE	CMS[1:0]		DIR	OPM	URS	UDIS	CEN
	Reset value																							0	0	0	0	0	0	0	0	0	0
0x004	TIM1_CR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OIS4	OIS3N	OIS3	OIS2N	OIS2	OIS1N	OIS1	TI1S	MMS [2:0]			CCDS	CCUS	Res.	CCPC
	Reset value																		0	0	0	0	0	0	0	0	0	0	0	0	0		0
0x008	TIM1_SMCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ETP	ECE	ETPS[1:0]		ETF[3:0]				MSM	TS[2:0]			OCCS	SMS [2:0]		
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x00C	TIM1_DIER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TDE	COMD	CC4DE	CC3DE	CC2DE	CC1DE	UDE	BIE	TIE	COMIE	CC4IE	CC3IE	CC2IE	CC1IE	UIE
	Reset value																		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x010	TIM1_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC4O	CC3O	CC2O	CC1O	Res.	BIF	TIF	COMIF	CC4IF	CC3IF	CC2IF	CC1IF	UIF
	Reset value																				0	0	0	0		0	0	0	0	0	0	0	0
0x014	TIM1_EGR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BG	TG	COMG	CC4G	CC3G	CC2G	CC1G	UG
	Reset value																									0	0	0	0	0	0	0	0
0x018	TIM1_CCMR1(output compare mode)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC2CE	OC2M [2:0]			OC2PE	CO2FE	CC2S [1:0]		OC1CE	OC1M [2:0]			OC1PE	OC1FE	CC1S [1:0]	
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x018	TIM1_CCMR1(Input Capture mode)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IC2F[3:0]				IC2PSC [1:0]		CC2S [1:0]		IC1F[3:0]				IC1PSC [1:0]		CC1S [1:0]	
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x01C	TIM1_CCMR2(output capture mode)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC4CE	OC4M [2:0]			OC4PE	CO4FE	CC4S [1:0]		OC3CE	OC3M [2:0]			OC3PE	OC3FE	CC3S [1:0]	
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

0x1C	TIM1_CCMR2(Input Capture mode)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IC4F[3:0]				IC4PSC [1:0]		CC4S [1:0]		IC3F[3:0]				IC3PSC [1:0]		CC3S [1:0]	
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x20	TIM1_CCER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC4P	CC4E	CC3NP	CC3NE	CC3P	CC3E	CC2NP	CC2NE	CC2P	CC2E	CC1NP	CC1NE	CC1P	CC1E
	Reset value																									0	0	0	0	0	0	0	0
0x24	TIM1_CNT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CNT[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x28	TIM1_PSC	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PSC[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x2C	TIM1_ARR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ARR[15:0]															
	Reset value																	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x30	TIM1_RCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REP[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x34	TIM1_CCR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR1[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x38	TIM1_CCR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR2[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x3C	TIM1_CCR3	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR3[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x40	TIM1_CCR4	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR4[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x44	TIM1_BDTR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MOE	AOE	BKP	BKE	OSSR	OSSI	LOCK [1:0]		DTG[7:0]							
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x48	TIM1_DCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBL[4:0]					Res.			DBA[4:0]				
	Reset value																				0	0	0	0	0				0	0	0	0	0

0x4C	TIM1_DMAR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DMAB[15:0]													
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0

18. General-purpose timer (TIM3)

18.1. TIM3 introduction

The general-purpose timer TIM3 consist of a 16-bit auto-reload counter driven by a programmable prescaler.

It may be used for a variety of purposes, including measuring the pulse lengths of input signals (input capture) or generating output waveforms (output compare and PWM).

Pulse lengths and waveform periods can be modulated from a few microseconds to several milliseconds using the timer prescaler and the RCC clock controller prescalers.

The timers are completely independent, and do not share any resources. They can be synchronized together.

18.2. TIM3 main features

General-purpose TIM3 timer features include:

- 16-bit (TIM3) up, down, up/down auto-reload counter.
- 16-bit programmable prescaler used to divide the counter clock frequency by any factor between 1 and 65535.
- Up to 4 independent channels for:
  - Input capture
  - Output compare
  - PWM generation (Edge- and Center-aligned modes)
  - One-pulse mode output
- Synchronization circuit to control the timer with external signals and to interconnect several timers.
- Interrupt/DMA generation on the following events:
  - Update: counter overflow/underflow, counter initialization (by software or internal/external trigger)
  - Trigger event (counter start, stop, initialization or count by internal/external trigger)
  - Input capture
  - Output compare
- Supports incremental (quadrature) encoder and hall-sensor circuitry for positioning purposes.
- Trigger input for external clock or cycle-by-cycle current management.

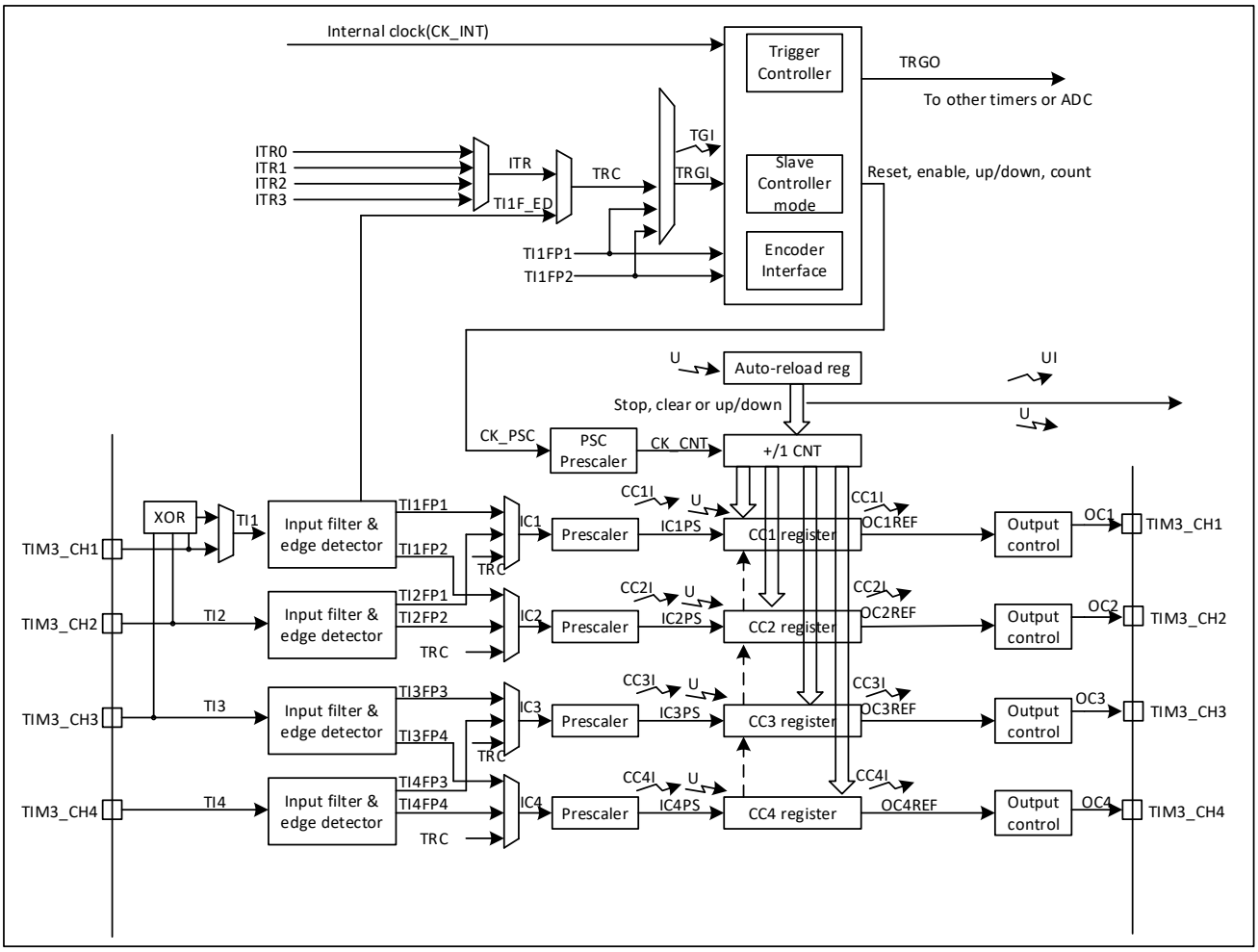

Figure 18-1 General-purpose timer block diagram (TIM3)

Notes:

Reg — Preload registers transferred to active registers on U event according to control bit

Event

Interrupt & DMA output

18.3. TIM3 functional description

18.3.1. Time-base unit

The main block of the programmable timer is a 16-bit counter with its related autoreload register. The counter can count up but also down or both up and down. The counter clock can be divided by a prescaler.

The counter, the auto-reload register and the prescaler register can be written or read by software. This is true even when the counter is running.

The time-base unit includes:

- Counter Register (TIM3_CNT)
- Prescaler Register (TIM3_PSC)
- Auto-Reload Register (TIM3_ARR)

The auto-reload register is preloaded. Writing to or reading from the auto-reload register accesses the preload register. The content of the preload register are transferred into the shadow register permanently or at each update event (UEV), depending on the auto-reload preload enable bit (ARPE) in TIM3_CR1 register. The update event is sent when the counter reaches the overflow (or underflow when downcounting) and if the UDIS bit equals 0 in the TIM3_CR1 register. It can also be generated by software.

The counter is clocked by the prescaler output CK_CNT, which is enabled only when the counter enable bit (CEN) in TIM3_CR1 register is set.

Note that the actual counter enable signal CNT_EN is set 1 clock cycle after CEN.

Prescaler description

The prescaler can divide the counter clock frequency by any factor between 1 and 65535. It is based on a 16-bit counter controlled through a 16-bit register (in the TIM3_PSC register). It can be changed on the fly as this control register is buffered. The new prescaler ratio is taken into account at the next update event.

The following figures give some examples of the counter behavior when the prescaler ratio is changed on the fly:

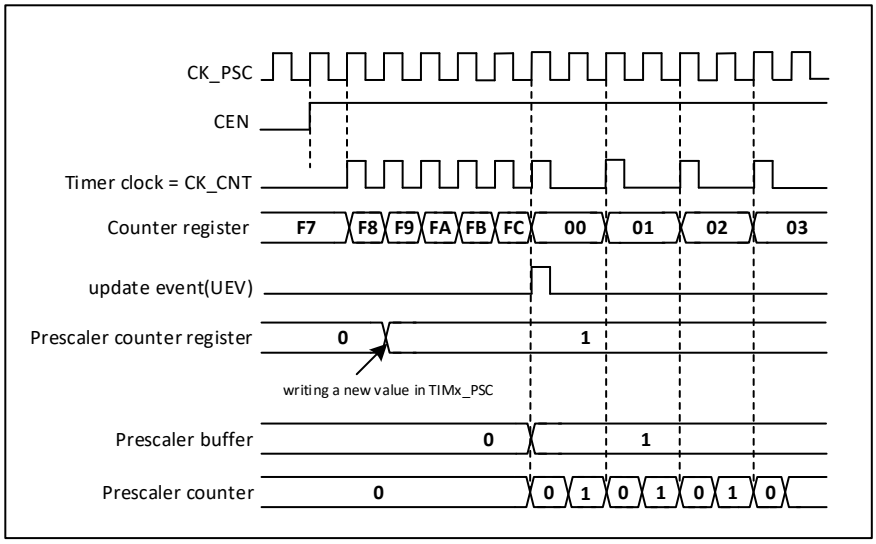

Figure 18-2 Counter timing diagram with prescaler division change from 1 to 2

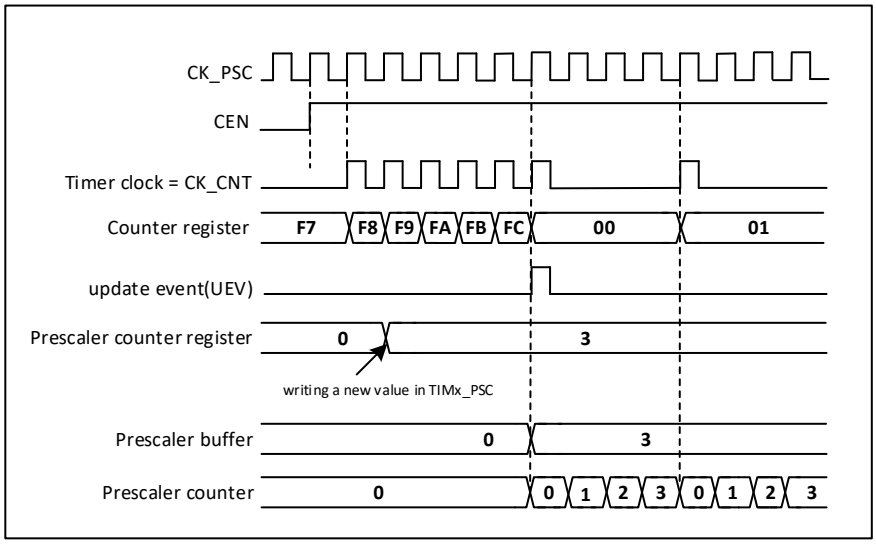

Figure 18-3 Counter timing diagram with prescaler division change from 1 to 4

18.3.2. Counter modes

Upcounting mode

In upcounting mode, the counter counts from 0 to the auto-reload value, then restarts from 0 and generates a counter overflow event.

An Update event can be generated by setting the UG bit in the TIM3_EGR register (by software or by using the slave mode controller).

The UEV event can be disabled by software by setting the UDIS bit in TIM3_CR1 register. This is to avoid updating the shadow registers while writing new values in the preload registers. Then no update event occurs until the UDIS bit has been written to 0. However, the counter restarts from 0, as well as the counter of the prescaler (but the prescale rate does not change). In addition, if the URS bit (update request selection) in TIM3_CR1 register is set, setting the UG bit generates an update event UEV but without setting the UIF flag (thus no interrupt or DMA request is sent). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

When an update event occurs, all the registers are updated and the update flag (UIF bit in TIM3_SR register) is set (depending on the URS bit):

- The auto-reload shadow register is updated with the preload value (TIM3_ARR).
- The buffer of the prescaler is reloaded with the preload value (content of the TIM3_PSC register).

The following figures show some examples of the counter behavior for different clock frequencies when TIM3_ARR = 0x36.

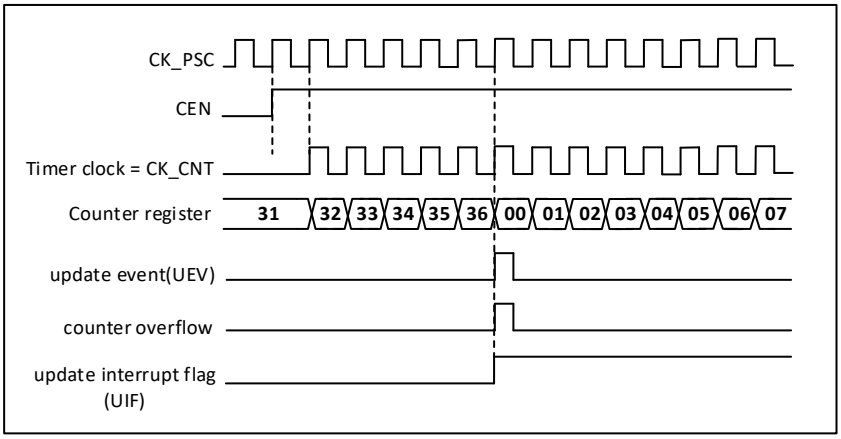

Figure 18-4 Counter timing diagram, internal clock divided by 1

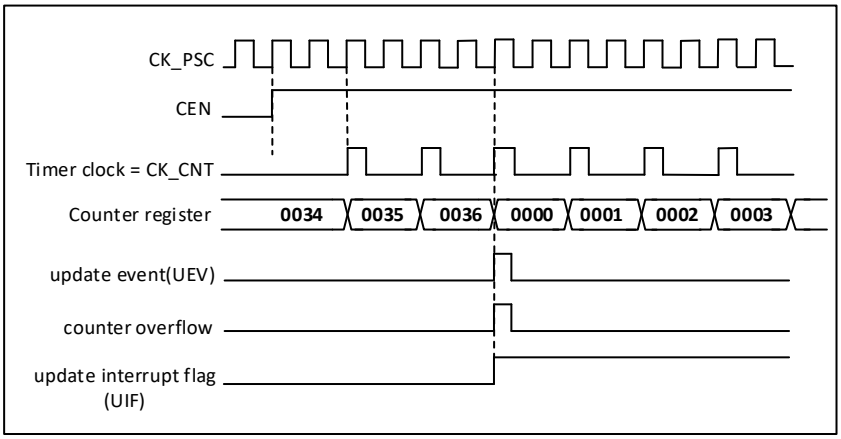

Figure 18-5 Counter timing diagram, internal clock divided by 2

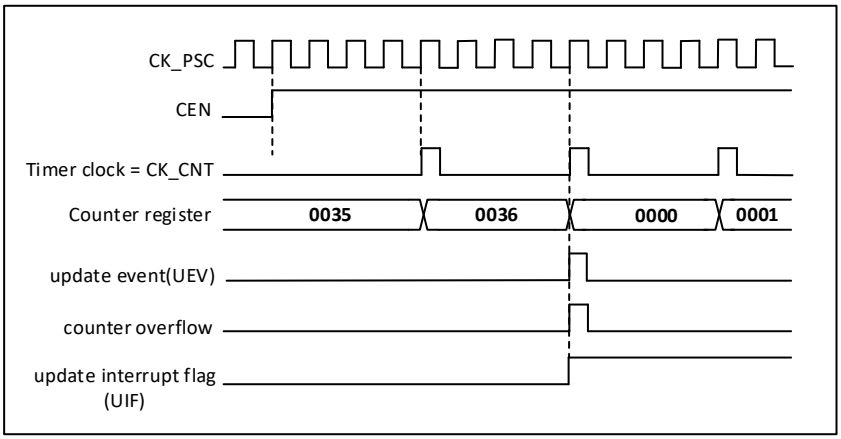

Figure 18-6 Counter timing diagram, internal clock divided by 4

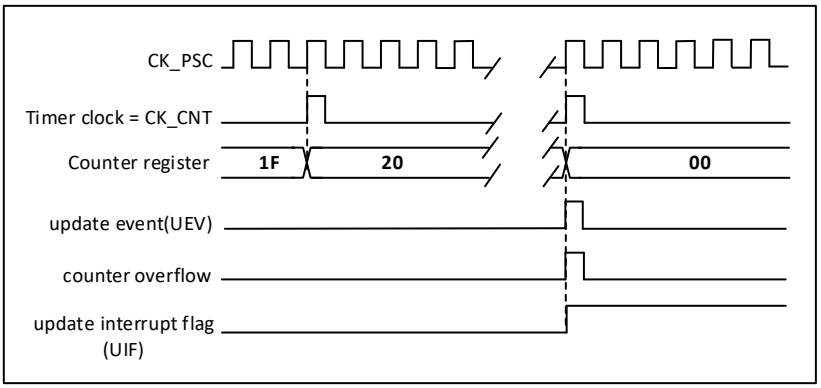

Figure 18-7 Counter timing diagram, internal clock divided by N

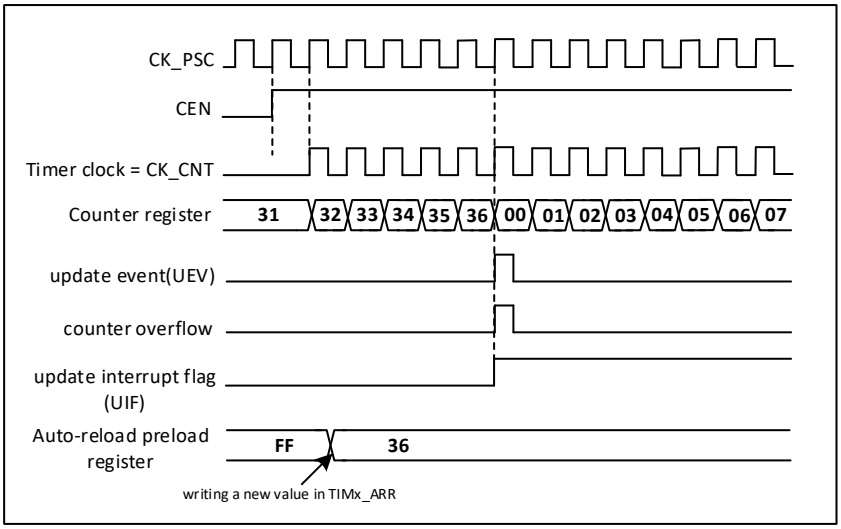

Figure 18-8 Counter timing diagram, Update event when ARPE = 0 (TIM3_ARR not preloaded)

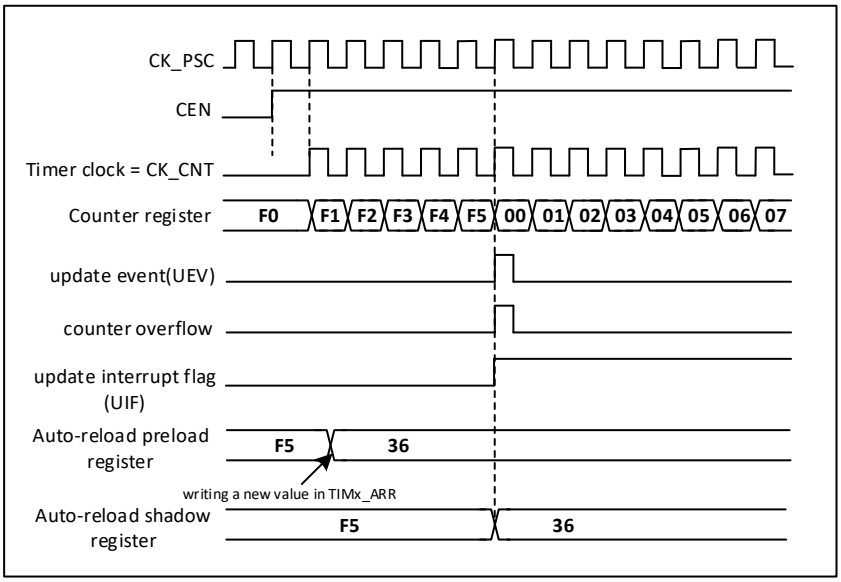

Figure 18-9 Counter timing diagram, Update event when ARPE = 1 (TIM3_ARR preloaded)

Downcounting mode

In downcounting mode, the counter counts from the auto-reload value down to 0, then restarts from the auto-reload value and generates a counter underflow event.

An Update event can be generated by setting the UG bit in the TIM3_EGR register (by software or by using the slave mode controller).

The UEV update event can be disabled by software by setting the UDIS bit in TIM3_CR1 register. This is to avoid updating the shadow registers while writing new values in the preload registers. Then no update event occurs until UDIS bit has been written to 0. However, the counter restarts from the current auto-reload value, whereas the counter of the prescaler restarts from 0 (but the prescale rate doesn't change).

In addition, if the URS bit (update request selection) in TIM3_CR1 register is set, setting the UG bit generates an update event UEV but without setting the UIF flag (thus no interrupt or DMA request is sent). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

When an update event occurs, all the registers are updated and the update flag (UIF bit in TIM3_SR register) is set (depending on the URS bit):

- The buffer of the prescaler is reloaded with the preload value (content of the TIM3_PSC register).
- he auto-reload active register is updated with the preload value (content of the TIM3_ARR register).

Note that the auto-reload is updated before the counter is reloaded, so that the next period is the expected one.

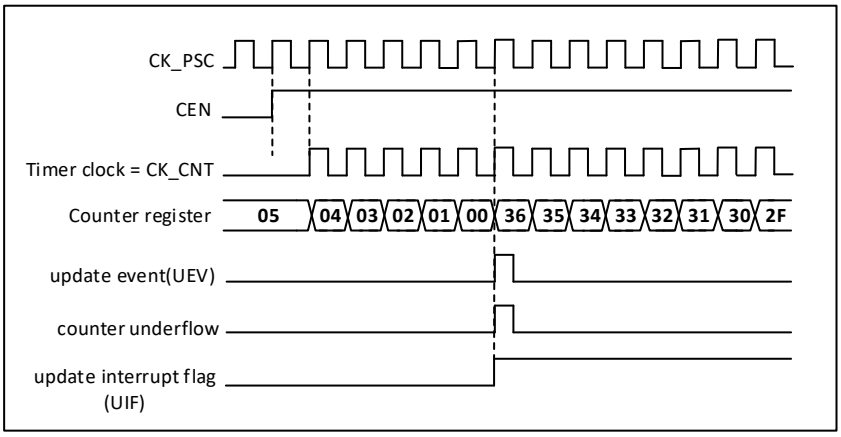

Figure 18-10 Counter timing diagram, internal clock divided by 1

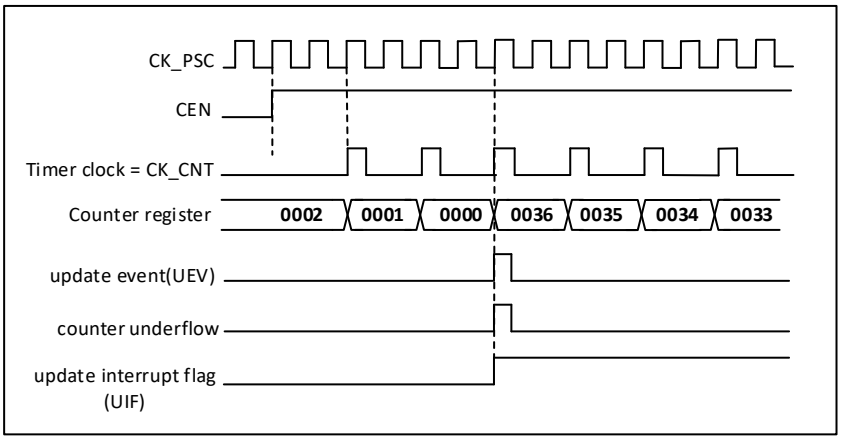

Figure 18-11 Counter timing diagram, internal clock divided by 2

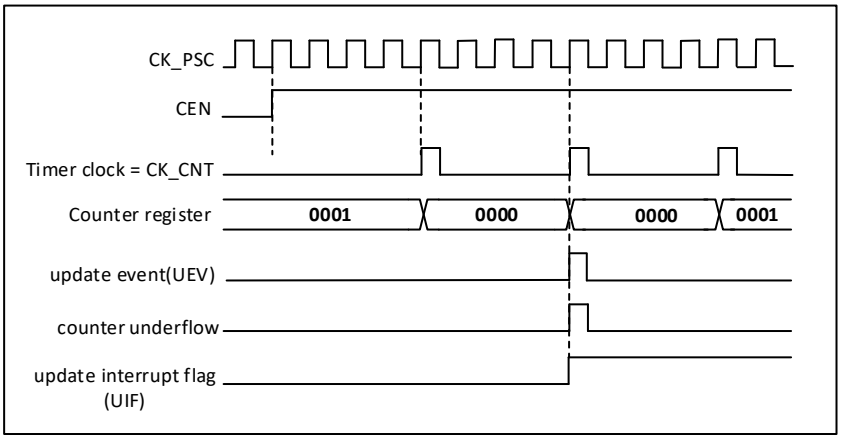

Figure 18-12 Counter timing diagram, internal clock divided by 4

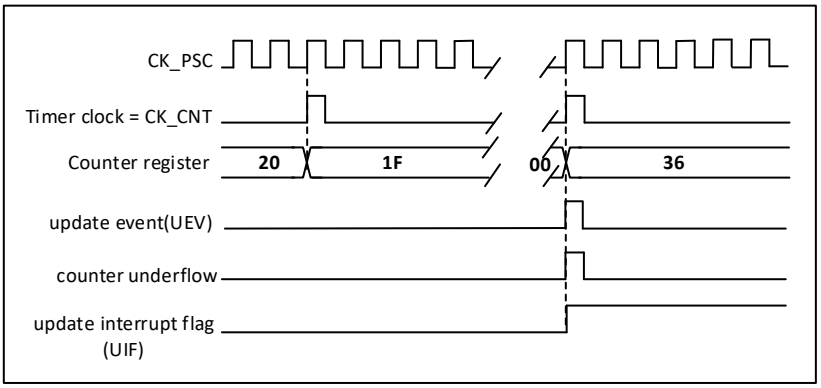

Figure 18-13 Counter timing diagram, internal clock divided by N

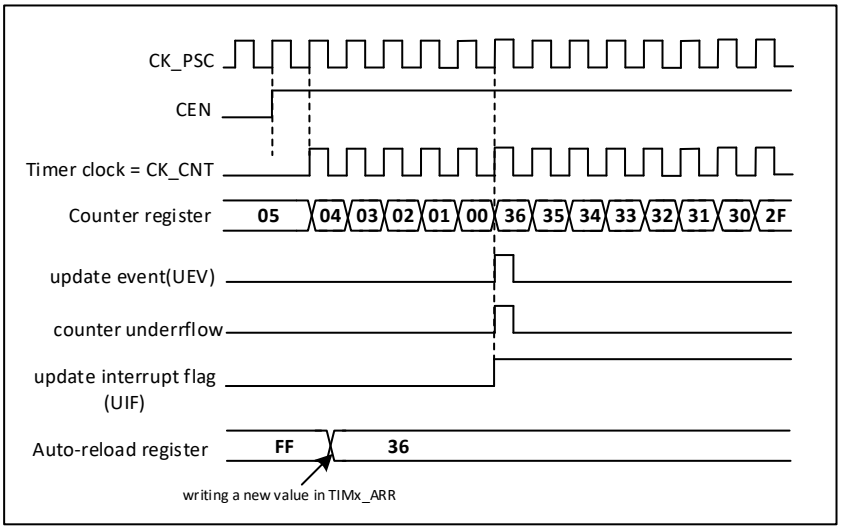

Figure 18-14 Counter timing diagram, Update event when repetition counter is not used

Center-aligned mode (up/down counting)

In center-aligned mode, the counter counts from 0 to the auto-reload value (content of the TIM3_ARR register) – 1, generates a counter overflow event, then counts from the autoreload value down to 1 and generates a counter underflow event. Then it restarts counting from 0.

Center-aligned mode is active when the CMS bits in TIM3_CR1 register are not equal to '00'. The Output compare interrupt flag of channels configured in output is set when: the counter counts down (Center aligned mode 1, CMS = “01”), the counter counts up (Center aligned mode 2, CMS = “10”) the counter counts up and down (Center aligned mode 3, CMS = “11”).

In this mode, the direction bit (DIR from TIM3_CR1 register) cannot be written. It is updated by hardware and gives the current direction of the counter.

The update event can be generated at each counter overflow and at each counter underflow or by setting the UG bit in the TIM3_EGR register (by software or by using the slave mode controller) also generates an update event. In this case, the counter restarts counting from 0, as well as the counter of the prescaler.

The UEV update event can be disabled by software by setting the UDIS bit in TIM3_CR1 register. This is to avoid updating the shadow registers while writing new values in the preload registers. Then no update event occurs until the UDIS bit has been written to 0. However, the counter continues counting up and down, based on the current auto-reload value.

In addition, if the URS bit (update request selection) in TIMx_CR1 register is set, setting the UG bit generates an update event UEV but without setting the UIF flag (thus no interrupt or DMA request is sent). This is to avoid generating both update and capture interrupt when clearing the counter on the capture event.

When an update event occurs, all the registers are updated and the update flag (UIF bit in TIM3_SR register) is set (depending on the URS bit):

- The buffer of the prescaler is reloaded with the preload value (content of the TIM3_PSC register).
- The auto-reload active register is updated with the preload value (content of the TIM3_ARR register).

Note that if the update source is a counter overflow, the autoreload is updated before the counter is reloaded, so that the next period is the expected one (the counter is loaded with the new value).

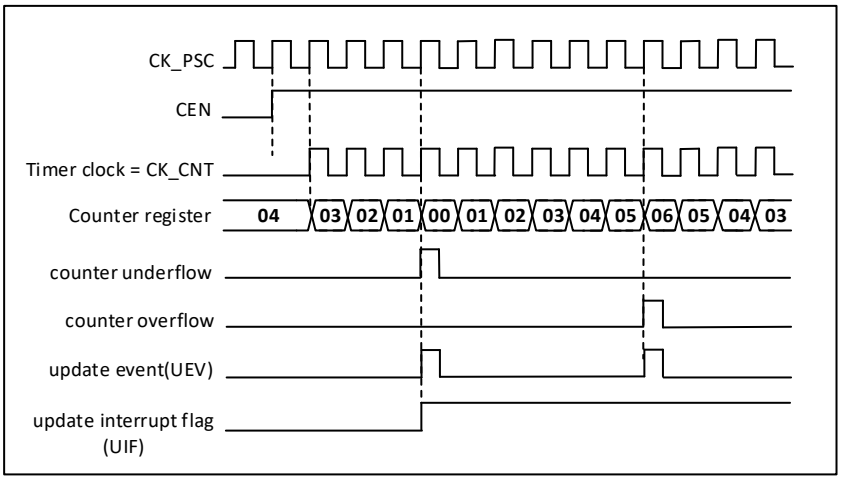

Figure 18-15 Counter timing diagram, internal clock divided by 1, TIM3_ARR = 0x6

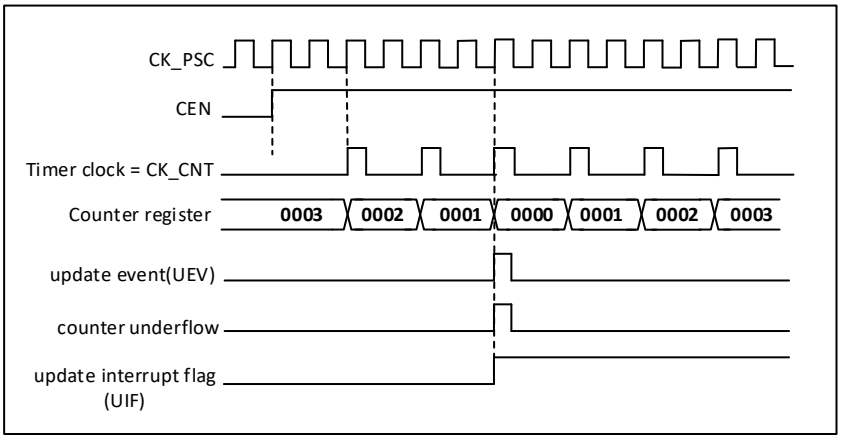

Figure 18-16 Counter timing diagram, internal clock divided by 2

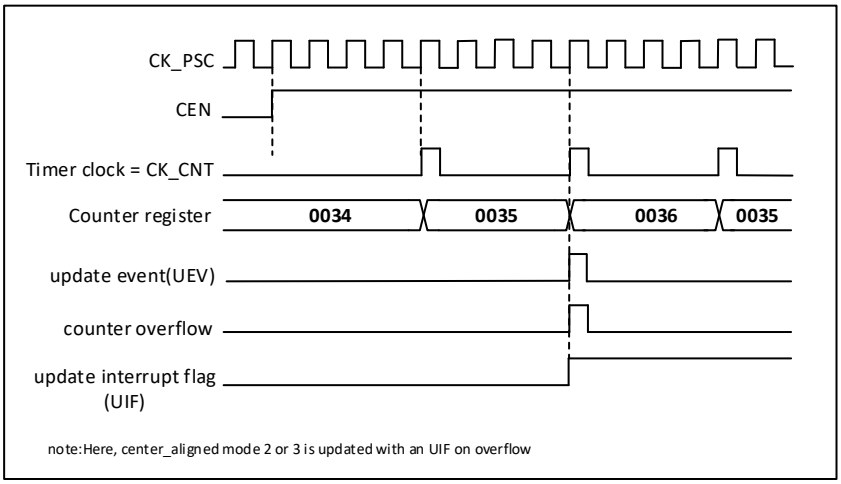

Figure 18-17 Counter timing diagram, internal clock divided by 4, TIM3_ARR = 0x36

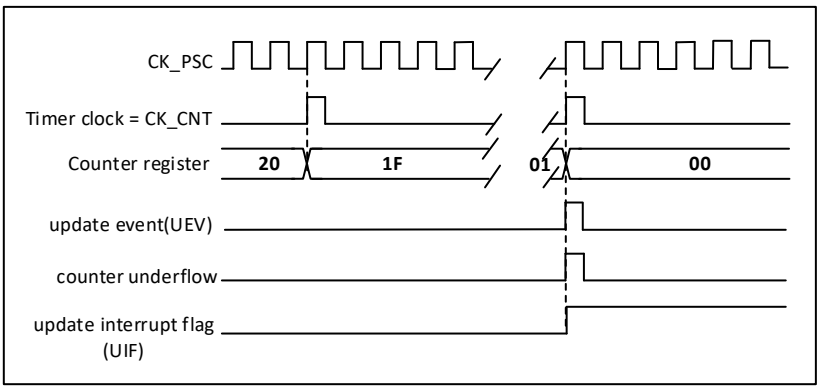

Figure 18-18 Counter timing diagram, internal clock divided by N

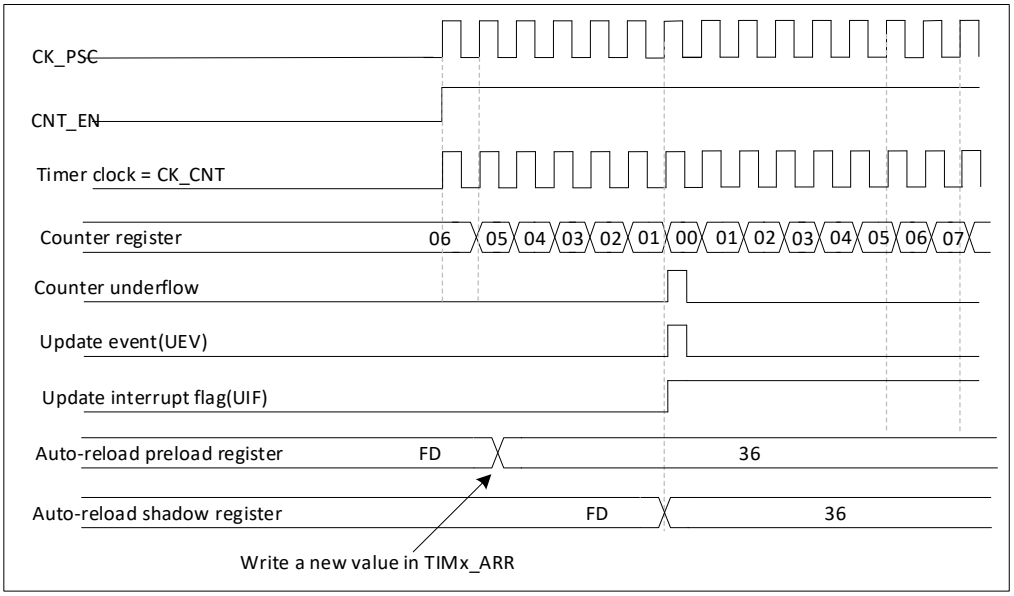

Figure 18-19 Counter timing diagram, Update event with ARPE = 1 (counter underflow)

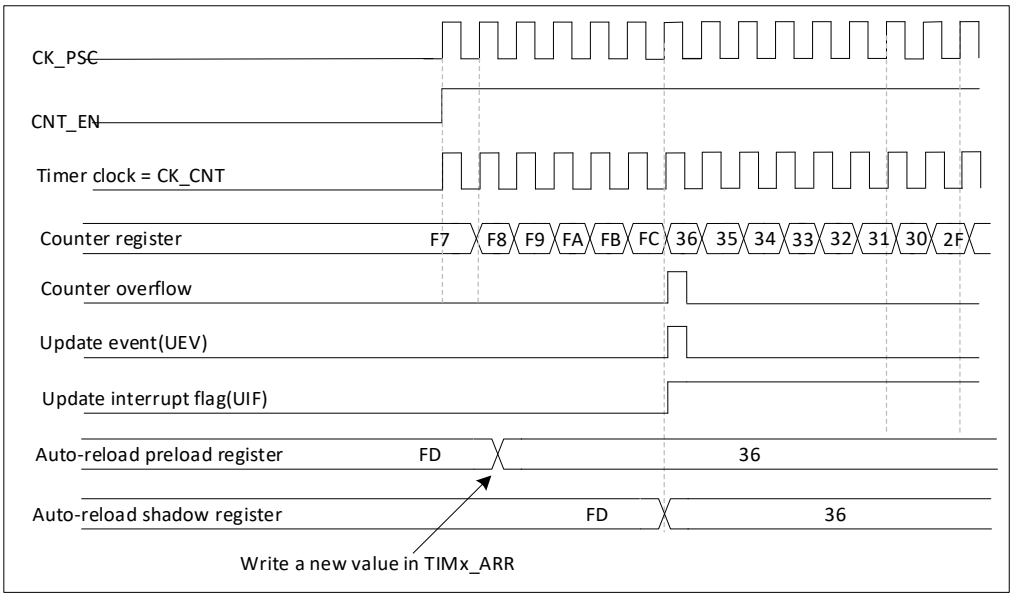

Figure 18-20 Counter timing diagram, Update event with ARPE = 1 (counter overflow)

18.3.3. Clock sources

The counter clock can be provided by the following clock sources:

- Internal clock (CK_INT)
- External clock mode1: external input pin (TIx)
- Internal trigger inputs (ITRx): using one timer as prescaler for another timer, for example, Timer 1 can be configured to act as a prescaler for Timer 3.

Internal clock source (CK_INT)

If the slave mode controller is disabled, then the CEN, DIR (in the TIM3_CR1 register) and UG bits (in the TIM3_EGR register) are actual control bits and can be changed only by software. As soon as the CEN bit is written to 1, the prescaler is clocked by the internal clock CK_INT.

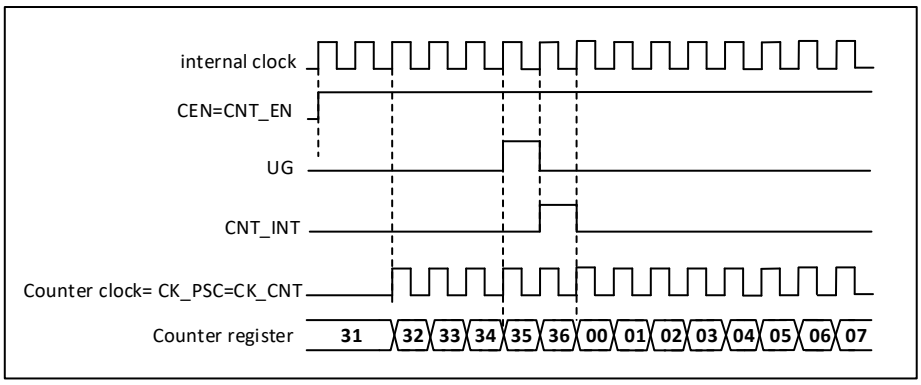

Figure 18-21 Control circuit in normal mode, internal clock divided by 1

External clock source mode 1

This mode is selected when SMS = 111 in the TIM3_SMCR register. The counter can count at each rising or falling edge on a selected input.

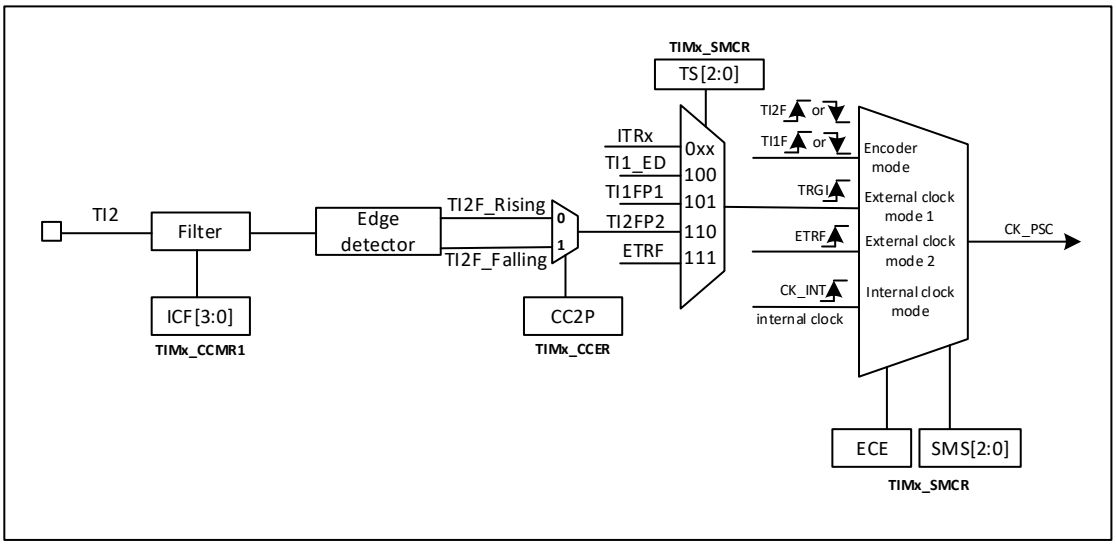

Figure 18-22 TI2 external clock connection example

For example, to configure the upcounter to count in response to a rising edge on the TI2 input, use the following procedure:

1. Configure channel 2 to detect rising edges on the TI2 input by writing CC2S = 01 in the TIM3_CCMR1 register.
2. Configure the input filter duration by writing the IC2F[3:0] bits in the TIM3_CCMR1 register (if no filter is needed, keep IC2F = 0000).
3. Select rising edge polarity by writing CC2P = 0 in the TIM3_CCER register.
4. Configure the timer in external clock mode 1 by writing SMS = 111 in the TIM3_SMCR register.
5. Select TI2 as the input source by writing TS = 110 in the TIM3_SMCR register.
6. Enable the counter by writing CEN = 1 in the TIM3_CR1 register.

Note: The capture prescaler is not used for triggering, so it does not need to be configured.

When a rising edge occurs on TI2, the counter counts once and the TIF flag is set.

The delay between the rising edge on TI2 and the actual clock of the counter is due to the resynchronization circuit on TI2 input.

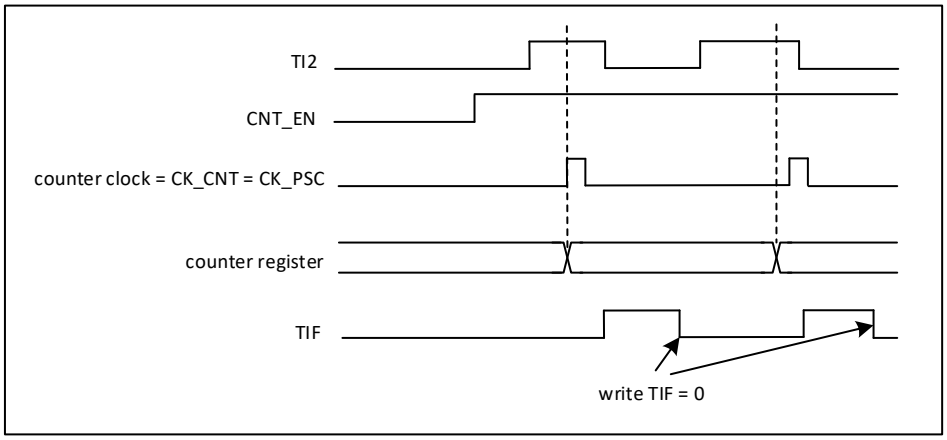

Figure 18-23 Control circuit in external clock mode 1

18.3.4. Capture/compare channels

Each Capture/Compare channel is built around a capture/compare register (including a shadow register), a input stage for capture (with digital filter, multiplexing and prescaler) and an output stage (with comparator and output control).

The input stage samples the corresponding TIx input to generate a filtered signal TIxF. Then, an edge detector with polarity selection generates a signal (TIxFPx) which can be used as trigger input by the slave mode controller or as the capture command. It is prescaled before the capture register (ICxPS).

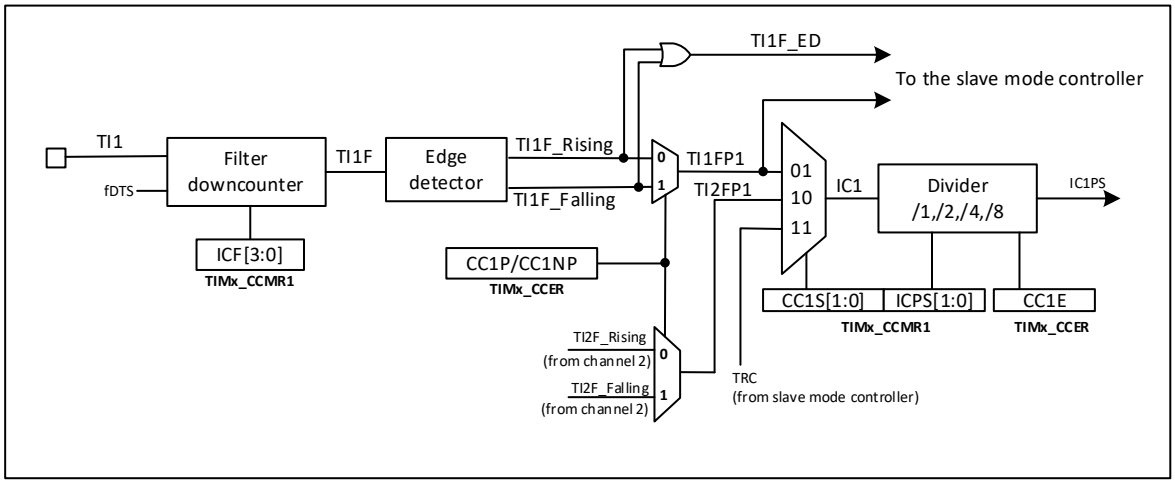

Figure 18-24 Capture/compare channel (example: channel 1 input stage)

The output stage generates an intermediate waveform which is then used for reference of OCxRef (active high). The polarity acts at the end of the chain.

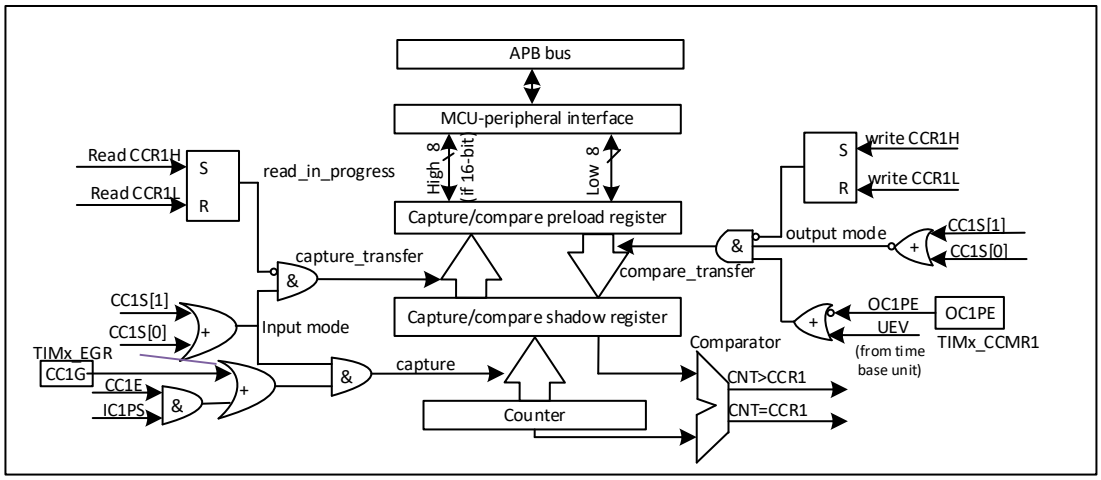

Figure 18-25 Capture/compare channel 1 main circuit

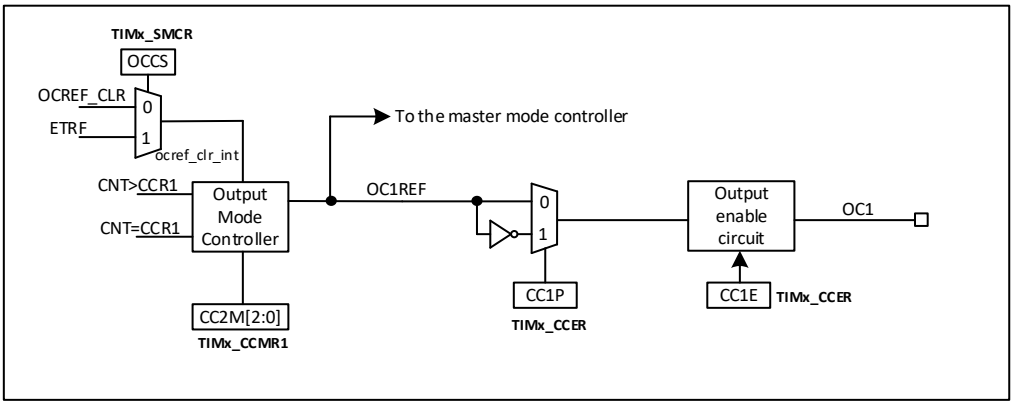

Figure 18-26 Output stage of capture/compare channel (channel 1)

The capture/compare block is made of one preload register and one shadow register. Write and read always access the preload register.

In capture mode, captures are actually done in the shadow register, which is copied into the preload register.

In compare mode, the content of the preload register is copied into the shadow register which is compared to the counter.

18.3.5. Input capture mode

In Input capture mode, the Capture/Compare Registers (TIM3_CCRx) are used to latch the value of the counter after a transition detected by the corresponding ICx signal. When a capture occurs, the corresponding CCxIF flag (TIM3_SR register) is set and an interrupt or a DMA request can be sent if they are enabled. If a capture occurs while the CCxIF flag was already high, then the over-capture flag CCxOF (TIM3_SR register) is set. CCxIF can be cleared by software by writing it to 0 or by reading the captured data stored in the TIM3_CCRx register. CCxOF is cleared when it is written with 0.

The following example shows how to capture the counter value in TIM3_CCR1 when TI1 input rises. To do this, use the following procedure:

- Select the active input: TIMx_CCR1 must be linked to the TI1 input, so write the CC1S bits to 01 in the TIM3_CCMR1 register. As soon as CC1S becomes different from 00, the channel is configured in input and the TIM3_CCR1 register becomes read-only.
- Program the appropriate input filter duration in relation with the signal connected to the timer (when the input is one of the TIx (ICxF bits in the TIM3_CCMRx register). Let’s imagine that, when toggling, the input

signal is not stable during at must 5 internal clock cycles. We must program a filter duration longer than these 5 clock cycles. We can validate a transition on TI1 when 8 consecutive samples with the new level have been detected (sampled at fDTS frequency). Then write IC1F bits to 0011 in the TIM3_CCMR1 register.

- Select the edge of the active transition on the TI1 channel by writing the CC1P and CC1NP bits to 0 in the TIM3_CCER register.
- Program the input prescaler. In our example, we wish the capture to be performed at each valid transition, so the prescaler is disabled (write IC1PS bits to 00 in the TIM3_CCMR1 register).
- Enable capture from the counter into the capture register by setting the CC1E bit in the TIM3_CCER register.
- If needed, enable the related interrupt request by setting the CC1IE bit in the TIM3_DIER register, and/or the DMA request by setting the CC1DE bit in the TIM3_DIER register.

When an input capture occurs:

- The TIM3_CCR1 register gets the value of the counter on the active transition.
- CC1IF flag is set (interrupt flag). CC1OF is also set if at least two consecutive captures occurred whereas the flag was not cleared.
- An interrupt is generated depending on the CC1IE bit.
- A DMA request is generated depending on the CC1DE bit.

In order to handle the overcapture, it is recommended to read the data before the overcapture flag. This is to avoid missing an overcapture which could happen after reading the flag and before reading the data.

Note: IC interrupt and/or DMA requests can be generated by software by setting the corresponding CCxG bit in the TIM3_EGR register.

18.3.6. PWM input mode

This mode is a particular case of input capture mode. The procedure is the same except:

- Two ICx signals are mapped on the same TIx input.
- The 2 ICx signals are active on edges with opposite polarity.
- One of the two TIxFP signals is selected as trigger input and the slave mode controller is configured in reset mode.

For example, one can measure the period (in TIM3_CCR1 register) and the duty cycle (in TIM3_CCR2 register) of the PWM applied on TI1 using the following procedure (depending on CK_INT frequency and prescaler value):

- Select the active input for TIM3_CCR1: write the CC1S bits to 01 in the TIM3_CCMR1 register (TI1 selected).
- Select the active polarity for TI1FP1 (used both for capture in TIM3_CCR1 and counter clear): write the CC1P to ‘0’(active on rising edge).
- Select the active input for TIM3_CCR2: write the CC2S bits to 10 in the TIM3_CCMR1 register (TI1 selected).
- Select the active polarity for TI1FP2 (used for capture in TIM3_CCR2): write the CC2P bit to ‘1’(active on falling edge).
- Select the valid trigger input: write the TS bits to 101 in the TIM3_SMCR register (TI1FP1 selected).
- Configure the slave mode controller in reset mode: write the SMS bits to 100 in the TIM3_SMCR register.
- Enable the captures: write the CC1E and CC2E bits to ‘1 in the TIM3_CCER register.

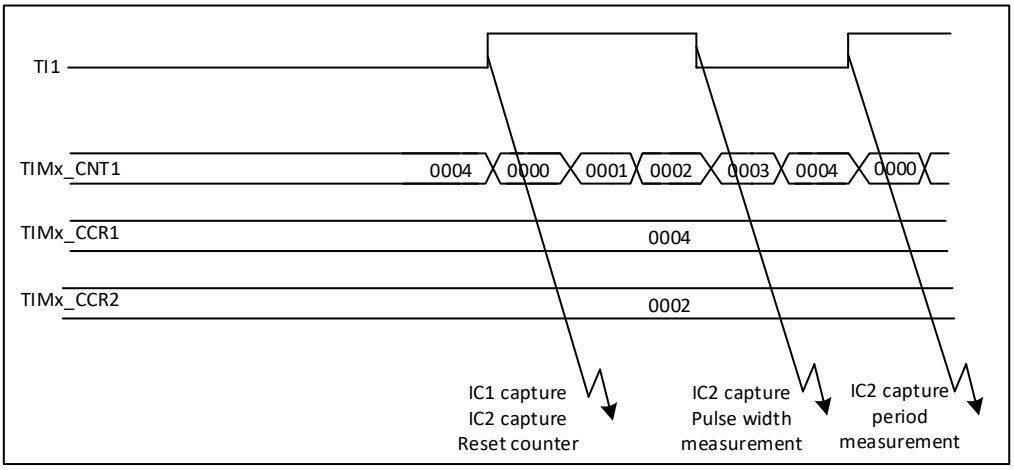

Figure 18-27 PWM input mode timing

18.3.7. Force output mode

In output mode (CCxS bits = 00 in the TIM3_CCMRx register), each output compare signal (OCxREF and then OCx) can be forced to active or inactive level directly by software, independently of any comparison between the output compare register and the counter. To force an output compare signal (OCxREF/OCx) to its active level, one just needs to write 101 in the OCxM bits in the corresponding TIM3_CCMRx register. Thus OCxREF is forced high(OCxREF is always active high) and OCx get opposite value to CCxP polarity bit.

Like, CCxP = 0 (OCx active high) = > OCx is forced to high level.

OCxREF signal can be forced low by writing the OCxM bits to 100 in the TIM3_CCMRx register.

Anyway, the comparison between the TIM3_CCRx shadow register and the counter is still performed and allows the flag to be set. Interrupt and DMA requests can be sent accordingly. This is described in the Output Compare Mode section.

18.3.8. Output compare mode

This function is used to control an output waveform or indicating when a period of time has elapsed. When a match is found between the capture/compare register and the counter, the output compare function:

- Assigns the corresponding output pin to a programmable value defined by the output compare mode (OCxM bits in the TIM3_CCMRx register) and the output polarity (CCxP bit in the TIM3_CCER register). The output pin can keep its level (OCxM = 000), be set active (OCxM = 001), be set inactive (OCxM = 010) or can toggle (OCxM = 011) on match.
- Sets a flag in the interrupt status register (CCxIF bit in the TIM3_SR register).
- Generates an interrupt if the corresponding interrupt mask is set (CCXIE bit in the TIM3_DIER register).
- Sends a DMA request if the corresponding enable bit is set (CCxDE bit in the TIM3_DIER register, CCDS bit in the TIM3_CR2 register for the DMA request selection).

The TIM3_CCRx registers can be programmed with or without preload registers using the OCxPE bit in the TIM3_CCMRx register.

In output compare mode, the update event UEV has no effect on OCxREF and OCx output. The timing resolution is one count of the counter. Output compare mode can also be used to output a single pulse (in One-pulse mode).

Procedure:

1. Select the counter clock (internal, external, prescaler).
2. Write the desired data in the TIM3_ARR and TIM3_CCRx registers.
3. Set the CCxIE bits if an interrupt request is to be generated.
4. Select the output mode. For example: one must write OCxM = 011, OCxPE = 0, CCxP = 0 and CCxE = 1 to toggle OCx output pin when CNT matches CCRx, CCRx preload is not used, OCx is enabled and active high.
5. Enable the counter by setting the CEN bit in the TIM3_CR1 register.

The TIM3_CCRx register can be updated at any time by software to control the output waveform, provided that the preload register is not enabled (OCxPE = 0, else TIM3_CCRx shadow register is updated only at the next update event UEV). An example is given.

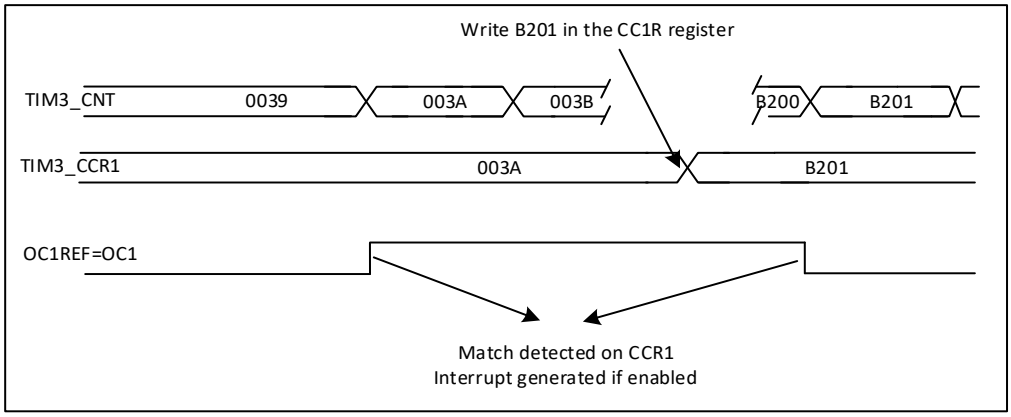

Figure 18-28 Output compare mode, toggle on OC1

18.3.9. PWM mode

Pulse width modulation mode allows to generate a signal with a frequency determined by the value of the TIM3_ARR register and a duty cycle determined by the value of the TIM3_CCRx register.

The PWM mode can be selected independently on each channel (one PWM per OCx output) by writing 110 (PWM mode 1) or ‘111 (PWM mode 2) in the OCxM bits in the TIM3_CCMRx register. The corresponding preload register must be enabled by setting the OCxPE bit in the TIM3_CCMRx register, and eventually the auto-reload preload register (in upcounting or center-aligned modes) by setting the ARPE bit in the TIMx_CR1 register. As the preload registers are transferred to the shadow registers only when an update event occurs, before starting the counter, all registers must be initialized by setting the UG bit in the TIM3_EGR register.

OCx polarity is software programmable using the CCxP bit in the TIMx_CCER register. It can be programmed as active high or active low. OCx output is enabled by the CCxE, CCxNE, MOE, OSSI and OSSR bit in the TIM3_CCER and TIM3_BDTR register. Refer to the TIM3_CCER register description for more details.

In PWM mode (1 or 2), TIM3_CNT and TIM3_CCRx are always compared to determine whether TIM3_CCRx≤TIM3_CNT or TIM3_CNT≤TIM3_CCRx (depending on the direction of the counter).

However, to comply with the OCREF_CLR functionality, the OCREF signal is asserted only:

1. When the result of the comparison changes.
2. When the output compare mode (OCxM bits in TIM3_CCMRx register) switches from the “frozen” configuration (no comparison, OCxM = ‘000) to one of the PWM modes (OCxM = ‘110 or ‘111).

This forces the PWM by software while the timer is running.

The timer is able to generate PWM in edge-aligned mode or center-aligned mode depending on the CMS bits in the TIM3_CR1 register.

PWM edge-aligned mode

- **Upcounting configuration**

Upcounting is active when the DIR bit in the TIMx_CR1 register is low. Refer to the following example of PWM Mode 1. The reference PWM signal OCxREF is high as long as TIM3_CNT < TIM3_CCRx else it becomes low. If the compare value in TIM3_CCRx is greater than the auto-reload value (in TIM3_ARR) then OCxREF is held at ‘1. If the compare value is 0 then OCxREF is held at ‘0. The following figure shows some edge-aligned PWM waveforms in an example where TIMx_ARR = 8.

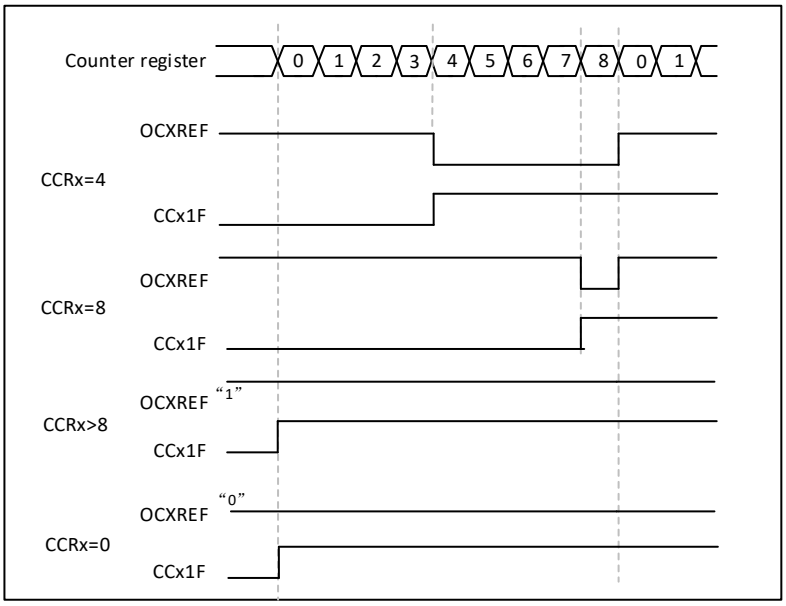

Figure 18-29 Edge-aligned PWM waveforms (ARR = 8)

Downcounting configuration

Downcounting is active when DIR bit in TIMx_CR1 register is high.

In PWM mode 1, the reference signal OCxREF is low as long as TIM3_CNT > TIM3_CCRx else it becomes high. If the compare value in TIM3_CCRx is greater than the auto-reload value in TIM3_ARR, then ocxref is held at ‘1’. 0% PWM is not possible in this mode.

PWM center-aligned mode

Center-aligned mode is active when the CMS bits in TIM3_CR1 register are different from‘00 (all the remaining configurations having the same effect on the OCxREF/OCx signals). The compare flag is set when the counter counts up, when it counts down or both when it counts up and down depending on the CMS bits configuration. The direction bit (DIR) in the TIM3_CR1 register is updated by hardware and must not be changed by software. Refer to the following example of the Center-aligned mode.

- TIMx_ARR = 8
- PWM mode is the PWM mode 1
- The flag is set when the counter counts down corresponding to the center-aligned mode 1 selected for CMS = 01 in TIM3_CR1 register.

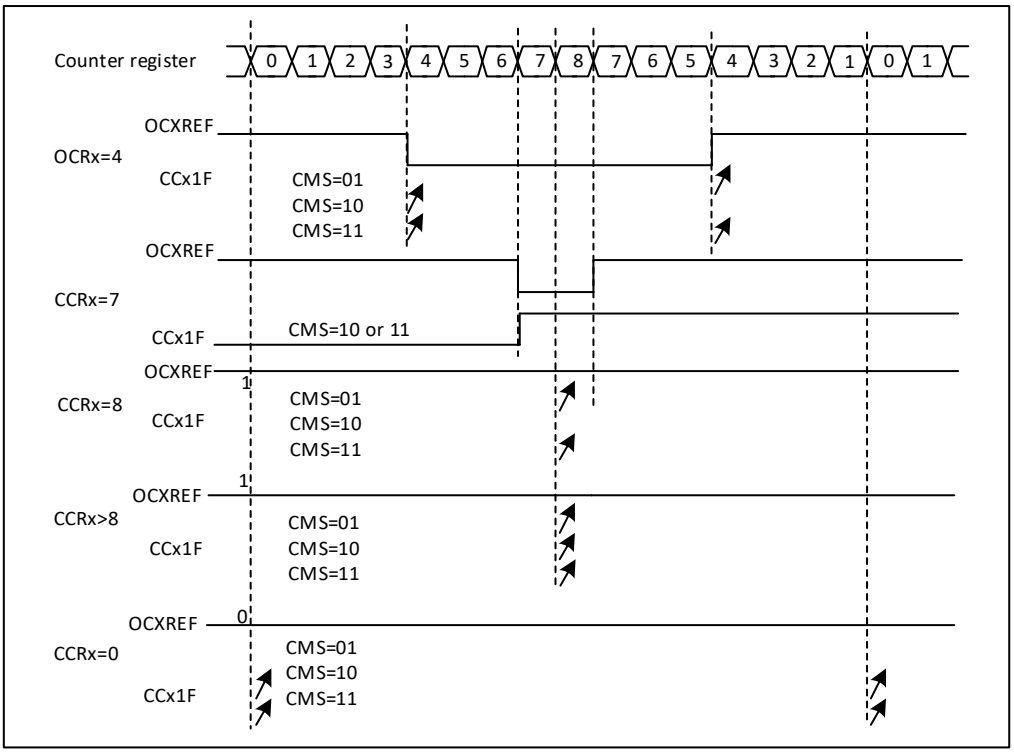

Figure 18-30 Center-aligned PWM waveforms (ARR = 8)

Hints on using center-aligned mode:

- When starting in center-aligned mode, the current up-down configuration is used. It means that the counter counts up or down depending on the value written in the DIR bit in the TIM3_CR1 register. Moreover, the DIR and CMS bits must not be changed at the same time by the software.
- Writing to the counter while running in center-aligned mode is not recommended as it can lead to unexpected results. In particular: ─ The direction is not updated if a value greater than the auto-reload value is written in the counter (TIM3_CNT > TIM3_ARR). For example, if the counter was counting up, it continues to count up. ─ The direction is updated if 0 or the TIM3_ARR value is written in the counter but no Update Event UEV is generated.
- The safest way to use center-aligned mode is to generate an update by software (setting the UG bit in the TIM3_EGR register) just before starting the counter and not to write the counter while it is running.

18.3.10. One-pulse mode

One-pulse mode (OPM) is a particular case of the previous modes. It allows the counter to be started in response to a stimulus and to generate a pulse with a programmable length after a programmable delay.

Starting the counter can be controlled through the slave mode controller. Generating the waveform can be done in output compare mode or PWM mode. One-pulse mode is selected by setting the OPM bit in the TIM3_CR1 register. This makes the counter stop automatically at the next update event UEV.

A pulse can be correctly generated only if the compare value is different from the counter initial value. Before starting (when the timer is waiting for the trigger), the configuration must be:

- Upcounting: CNT < CCRx ≤ ARR (in particular, 0 < CCRx),
- Downcounting: CNT > CCRx.

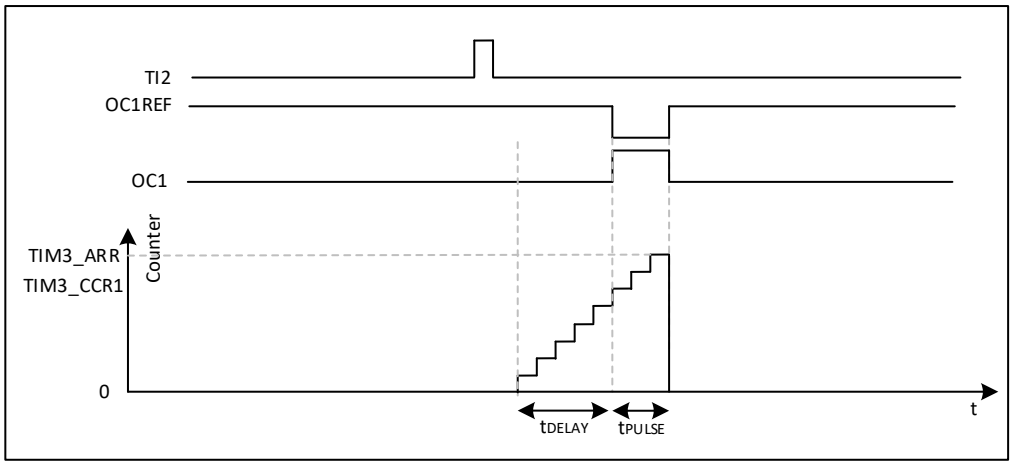

Figure 18-31 Example of one-pulse mode

For example one may want to generate a positive pulse on OC1 with a length of t_{PLUSE} and after a delay of t_{DELAY} as soon as a positive edge is detected on the TI2 input pin.

Use TI2FP2 as trigger 1:

- Map TI2FP2 on TI2 by writing CC2S = 01 in the TIM3_CCMR1 register.
- TI2FP2 must detect a rising edge, write CC2P = 0 in the TIM3_CCER register.
- Configure TI2FP2 as trigger for the slave mode controller (TRGI) by writing TS = 110 in the TIM3_SMCR register.
- TI2FP2 is used to start the counter by writing SMS to ‘110 in the TIM3_SMCR register (trigger mode).

The OPM waveform is defined by writing the compare registers (taking into account the clock frequency and the counter prescaler).

- The t_{DELAY} is defined by the value written in the TIM3_CCR1 register.
- The t_{PULSE} is defined by the difference between the auto-reload value and the compare value (TIM3_ARR - TIM3_CCR1 + 1).
- When the user to build a waveform with a transition from ‘0’ to ‘1’ when a compare match occurs and a transition from ‘1 to ‘0 when the counter reaches the auto-reload value. To do this PWM mode 2 must be enabled by writing OC1M = 111 in the TIM3_CCMR1 register. Optionally the preload registers can be enabled by writing OC1PE = 1 in the TIM3_CCMR1 register and ARPE in the TIM3_CR1 register. In this case one has to write the compare value in the TIM3_CCR1 register, the auto-reload value in the TIM3_ARR register, generate an update by setting the UG bit and wait for external trigger event on TI2. CC1P is written to ‘0 in this example.

In the example, the DIR and CMS bits in the TIM3_CR1 register should be low.

Since only 1 pulse (Single mode) is needed, a 1 must be written in the OPM bit in the TIM3_CR1 register to stop the counter at the next update event (when the counter rolls over from the auto-reload value back to 0).

Particular case: OCx fast enable

In One-pulse mode, the edge detection on TIx input set the CEN bit which enables the counter. Then the comparison between the counter and the compare value makes the output toggle. But several clock cycles are needed for these operations and it limits the minimum delay t_{DELAY}.

If one wants to output a waveform with the minimum delay, the OCxFE bit can be set in the TIM3_CCMRx register. Then OCxRef (and OCx) are forced in response to the stimulus, without taking in account the comparison. Its new level is the same as if a compare match had occurred. OCxFE acts only if the channel is configured in PWM1 or PWM2 mode.

18.3.11. Encoder interface mode

To select Encoder Interface mode write SMS = ‘001 in the TIM3_SMCR register if the counter is counting on TI2 edges only, SMS = 010 if it is counting on TI1 edges only and SMS = 011 if it is counting on both TI1 and TI2 edges.

Select the TI1 and TI2 polarity by programming the CC1P and CC2P bits in the TIM3_CCER register. When needed, the input filter can be programmed as well.

The two inputs TI1 and TI2 are used to interface to an incremental encoder. Refer to Table73, The counter is clocked by each valid transition on TI1FP1 or TI2FP2 assuming that it is enabled (CEN bit in TIM3_CR1 register written to ‘1). TI1 and TI2 after input filter and polarity selection, TI1FP1 = TI1 if not filtered and not inverted, TI2FP2 = TI2 if not filtered and not inverted. The sequence of transitions of the two inputs is evaluated and generates count pulses as well as the direction signal. Depending on the sequence the counter counts up or down, the DIR bit in the TIM3_CR1 register is modified by hardware accordingly. The DIR bit is calculated at each transition on any input (TI1 or TI2), whatever the counter is counting on TI1 only, TI2 only or both TI1 and TI2.

Encoder interface mode acts simply as an external clock with direction selection. This means that the counter just counts continuously between 0 and the auto-reload value in the TIM3_ARR register (0 to ARR or ARR down to 0 depending on the direction). So the TIM3_ARR must be configured before starting. In the same way, the capture, compare, prescaler, trigger output features continue to work as normal. In this mode, the counter is modified automatically following the speed and the direction of the incremental encoder and its content, therefore, always represents the encoder’s position. The count direction correspond to the rotation direction of the connected sensor. The table summarizes the possible combinations, assuming TI1 and TI2 do not switch at the same time.

Table 18-1 Counting direction versus encoder signals

Active edge	Level on opposite signal (TI1FP1 for TI2, TI2FP2 for TI1)	TI1FP1 signal		TI2FP2 signal	
		Rising	Falling	Rising	Falling
Counting on TI1 only	High	Down	Up	No count	No count
	Low	Up	Down	No count	No count
Counting on TI2 only	High	No count	No count	Up	Down
	Low	No count	No count	Down	Up
Counting on TI1 and TI2	High	Up	Up	Up	Down
	Low	Down	Down	Down	Up

An external incremental encoder can be connected directly to the MCU without external interface logic. However, comparators are normally be used to convert the encoder’s differential outputs to digital signals. This greatly increases noise immunity. The third encoder output which indicate the mechanical zero position, may be connected to an external interrupt input and trigger a counter reset.

Figure 18-32 gives an example of counter operation, showing count signal generation and direction control. It also shows how input jitter is compensated where both edges are selected. This might occur if the sensor is positioned near to one of the switching points. For this example we assume that the configuration is the following:

- CC1S = 01 (TIM3_CCMR1 register, TI1FP1 mapped on TI1)

- CC2S = 01 (TIM3_CCMR2 register, TI2FP2 mapped on TI2)
- CC1P = 0, CC1NP = ‘0’ (TIM3_CCER register, TI1FP1 noninverted, TI1FP1 = TI1)
- CC2P = 0, CC2NP = ‘0’ (TIM3_CCER register, TI2FP2 noninverted, TI2FP2 = TI2)
- SMS = 011 (TIM3_SMCR register, both inputs are active on both rising and falling edges)
- CEN = 1 (TIM3_CR1 register, Counter is enabled)

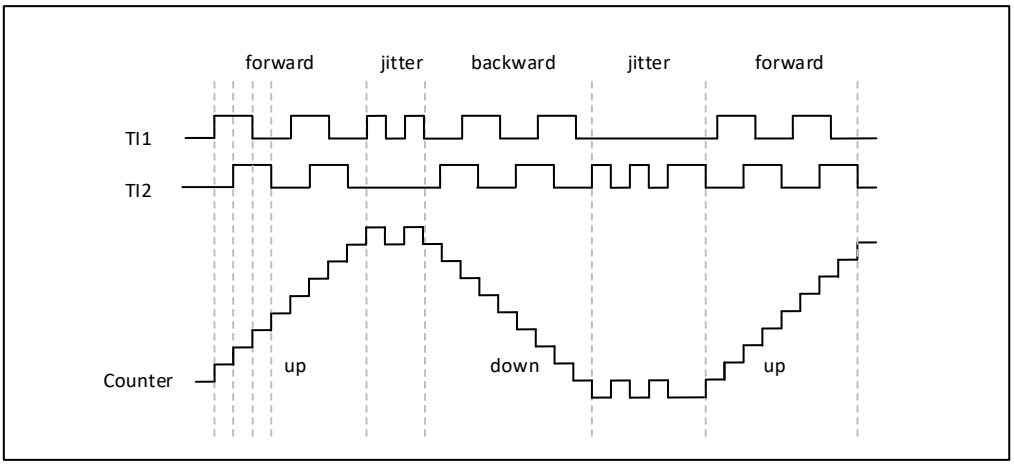

Figure 18-32 Example of counter operation in encoder interface mode

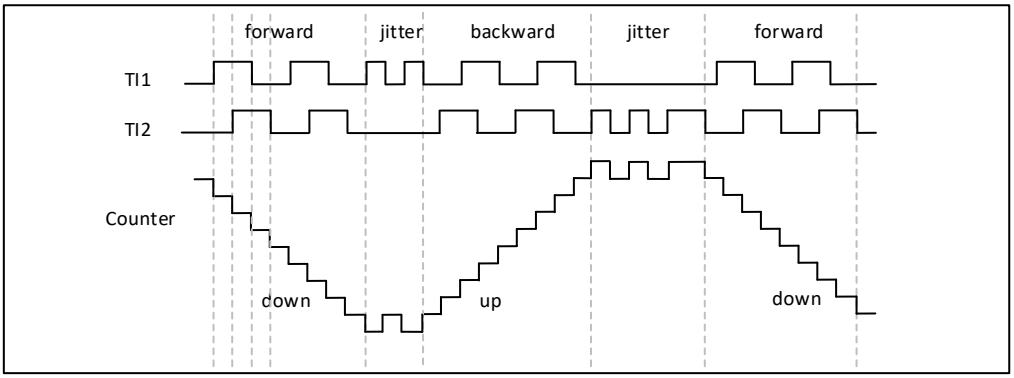

Figure 18-33 Example of encoder interface mode with TI1FP1 polarity inverted

The timer, when configured in Encoder Interface mode provides information on the sensor’s current position. Dynamic information can be obtained (speed, acceleration, deceleration) by measuring the period between two encoder events using a second timer configured in capture mode. The output of the encoder which indicates the mechanical zero can be used for this purpose. Depending on the time between two events, the counter can also be read at regular times. This can be done by latching the counter value into a third input capture register if available (then the capture signal must be periodic and can be generated by another timer). when available, it is also possible to read its value through a DMA request generated by a Real-Time clock.

18.3.12. Timer input XOR function

The TI1S bit in the TIM1_CR2 register, allows the input filter of channel 1 to be connected to the output of a XOR gate, combining the three input pins TIM3_CH1 to TIM3_CH3.

The XOR output can be used with all the timer input functions such as trigger or input capture.

An example of this feature used to interface Hall sensors.

18.3.13. Timers and external trigger synchronization

The TIM3 Timers can be synchronized with an external trigger in several modes: Reset mode, Gated mode and Trigger mode.

Slave mode: Reset mode

The counter and its prescaler can be reinitialized in response to an event on a trigger input. Moreover, if the URS bit from the TIM3_CR1 register is low, an update event UEV is generated. Then all the preloaded registers (TIM3_ARR, TIM3_CCRx) are updated.

In the following example, the upcounter is cleared in response to a rising edge on TI1 input:

- Configure the channel 1 to detect rising edges on TI1. Configure the input filter duration (in this example, we do not need any filter, so we keep IC1F = 0000). The capture prescaler is not used for triggering, so it does not need to be configured. The CC1S bits select the input capture source only, CC1S = 01 in the TIM3_CCMR1 register. Write CC1P = 0 in TIM3_CCER register to validate the polarity (and detect rising edges only).
- Configure the timer in reset mode by writing SMS = 100 in TIM3_SMCR register. Select TI1 as the input source by writing TS = 101 in TIM3_SMCR register.
- Start the counter by writing CEN = 1 in the TIM3_CR1 register.

The counter starts counting on the internal clock, then behaves normally until TI1 rising edge. When TI1 rises, the counter is cleared and restarts from 0. In the meantime, the trigger flag is set (TIF bit in the TIM3_SR register) and an interrupt request, or a DMA request can be sent if enabled (depending on the TIE and TDE bits in TIM3_DIER register).

The following figure shows this behavior when the auto-reload register TIM3_ARR = 0x36. The delay between the rising edge on TI1 and the actual reset of the counter is due to the resynchronization circuit on TI1 input.

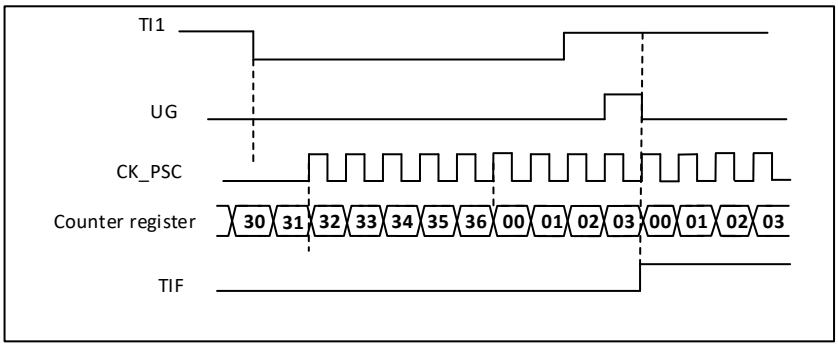

Figure 18-34 Control circuit in reset mode

Slave mode: Gated mode

The counter can be enabled depending on the level of a selected input.

In the following example, the upcounter counts only when TI1 input is low:

- Configure the channel 1 to detect low levels on TI1. Configure the input filter duration (in this example, we do not need any filter, so we keep IC1F = 0000). The capture prescaler is not used for triggering, so it does not need to be configured. The CC1S bits select the input capture source only, CC1S = 01 in TIM3_CCMR1 register. Write CC1P = 1 in TIM3_CCER register to validate the polarity (and detect low level only).
- Configure the timer in gated mode by writing SMS = 101 in TIM3_SMCR register. Select TI1 as the input source by writing TS = 101 in TIM3_SMCR register.

- Enable the counter by writing CEN = 1 in the TIM3_CR1 register (in gated mode, the counter doesn't start if CEN = 0, whatever is the trigger input level).

The counter starts counting on the internal clock as long as TI1 is low and stops as soon as TI1 becomes high. The TIF flag in the TIMx_SR register is set both when the counter starts or stops.

The delay between the rising edge on TI1 and the actual stop of the counter is due to the resynchronization circuit on TI1 input.

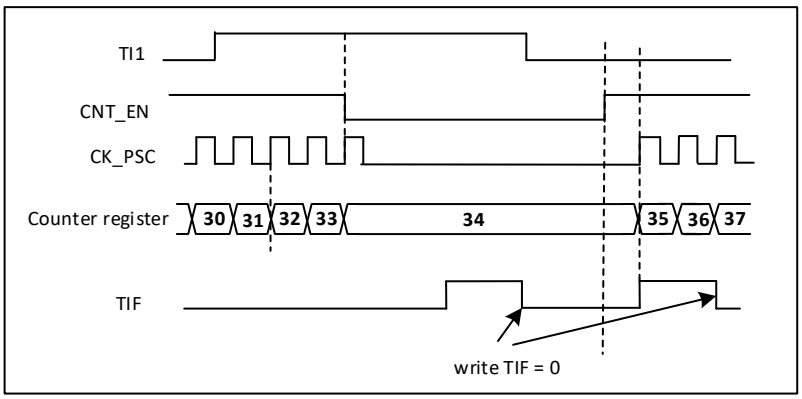

Figure 18-35 Control circuit in gated mode

Slave mode: Trigger mode

The counter can start in response to an event on a selected input.

In the following example, the upcounter starts in response to a rising edge on TI2 input:

- Configure the channel 2 to detect rising edges on TI2. Configure the input filter duration (in this example, we do not need any filter, so we keep IC2F = 0000). The capture prescaler is not used for triggering, so it does not need to be configured. CC2S bits are selecting the input capture source only, CC2S = 01 in TIM3_CCMR1 register. Write CC2P = 1 and CC2NP = 0 in TIM3_CCER register to validate the polarity (and detect low level only).
- Configure the timer in trigger mode by writing SMS = 110 in TIM3_SMCR register. Select TI2 as the input source by writing TS = 110 in TIM3_SMCR register.

When a rising edge occurs on TI2, the counter starts counting on the internal clock and the TIF flag is set. The delay between the rising edge on TI2 and the actual start of the counter is due to the resynchronization circuit on TI2 input.

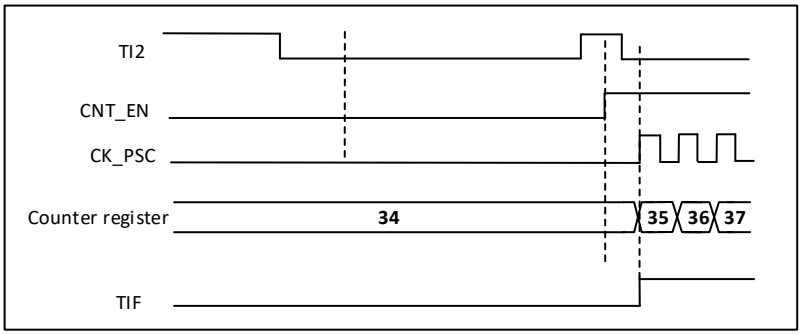

Figure 18-36 Control circuit in trigger mode

Slave mode: External Clock mode 2 + trigger mode

The external clock mode 2 can be used in addition to another slave mode (except external clock mode 1 and encoder mode). In this case, the ETR signal is used as external clock input, and another input can be selected as trigger input when operating in reset mode, gated mode or trigger mode. It is recommended not to select ETR as TRGI through the TS bits of TIM3_SMCR register.

In the following example, the upcounter is incremented at each rising edge of the ETR signal as soon as a rising edge of TI1 occurs:

1. Configure the external trigger input circuit by programming the TIMx_SMCR register as follows:
   - ETF = 0000: no filter
   - ETPS = 00: prescaler disabled
   - ETP = 0: detection of rising edges on ETR and ECE = 1 to enable the external clock mode 2.
2. Configure the channel 1 as follows, to detect rising edges on TI:
   - IC1F = 0000: no filter.
   - The capture prescaler is not used for triggering and does not need to be configured.
   - CC1S = 01 in TIM3_CCMR1 register to select only the input capture source
   - CC1P = 0 in TIM3_CCER register to validate the polarity (and detect rising edge only).
3. Configure the timer in trigger mode by writing SMS = 110 in TIM3_SMCR register. Select TI1 as the input source by writing TS = 101 in TIM3_SMCR register.

The delay between the rising edge of the ETR signal and the actual reset of the counter is due to the resynchronization circuit on ETRP input.

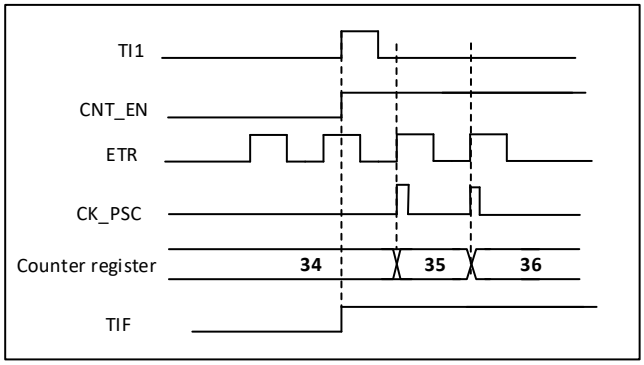

Figure 18-37 Control circuit in external clock mode 2 + trigger mode

18.3.14. Timer synchronization

The TIMx timers are linked together internally for timer synchronization or chaining. When one Timer is configured in Master Mode, it can reset, start, stop or clock the counter of another Timer configured in Slave Mode.

The following figure presents an overview of the trigger selection and the master mode selection blocks.

Using one timer as prescaler for another

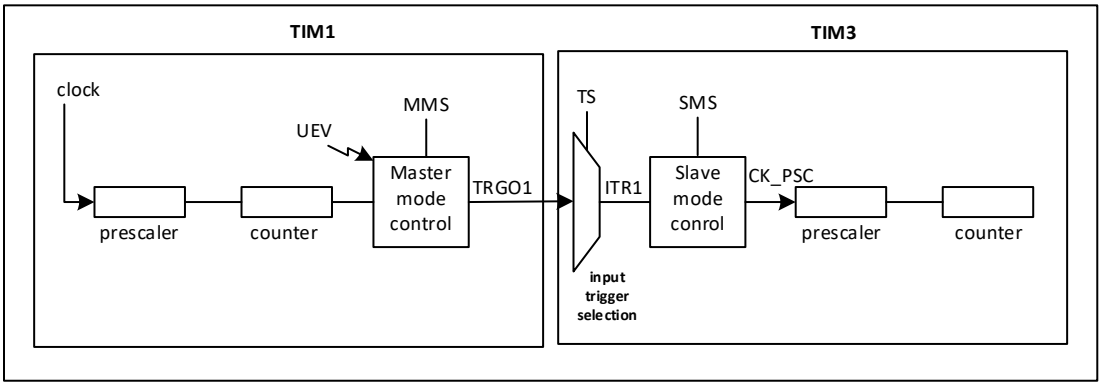

Figure 18-38 Master/Slave timer example

For example, Timer 1 can be configured to act as a prescaler for Timer 3. To do this:

- Configure Timer 1 in master mode so that it outputs a periodic trigger signal on each update event UEV. If MMS = 010 is written in the TIM1_CR2 register, a rising edge is output on TRGO1 each time an update event is generated.
- To connect the TRGO1 output of Timer 1 to Timer 3, Timer 3 must be configured in slave mode using ITR1 as internal trigger. This is selected through the TS bits in the TIM3_SMCR register (writing TS = 000).
- Then the Timer3's slave mode controller should be configured in external clock mode 1 (write SMS = 111 in the TIM3_SMCR register). This causes Timer 3 to be clocked by the rising edge of the periodic Timer 1 trigger signal (which correspond to the timer 1 counter overflow).
- Finally both timers must be enabled by setting their respective CEN bits within their respective TIMx_CR1 registers. Make sure to enable Timer3 before enabling Timer1.

Note: If OCx is selected on Timer 1 as trigger output (MMS = 1xx), its rising edge is used to clock the counter of timer 2.

Using one timer to enable another timer

In this example, we control the enable of Timer 3 with the output compare 1 of Timer 1. Refer to above Figure for connections. Timer 3 counts on the divided internal clock only when OC1REF of Timer 1 is high. Both counter clock frequencies are divided by 3 by the prescaler compared to CK_INT ($f_{CK\_CNT} = f_{CK\_INT}/3$).

- Configure Timer 1 master mode to enable the slave timer(MMS = 001 in the TIM1_CR2 register).
- Configure the Timer 1 OC1REF waveform (TIM1_CCMR1 register).
- Configure Timer 3 to get the input trigger from Timer 1 (TS = 000 in the TIM3_SMCR register).
- Configure Timer 3 in gated mode (SMS = 101 in TIM3_SMCR register).
- Enable Timer 3 by writing ‘1 in the CEN bit (TIM3_CR1 register).
- Start Timer 1 by writing ‘1 in the CEN bit (TIM1_CR1 register).

Note: The counter 3 clock is not synchronized with counter 1, this mode only affects the Timer 3 counter enable signal.

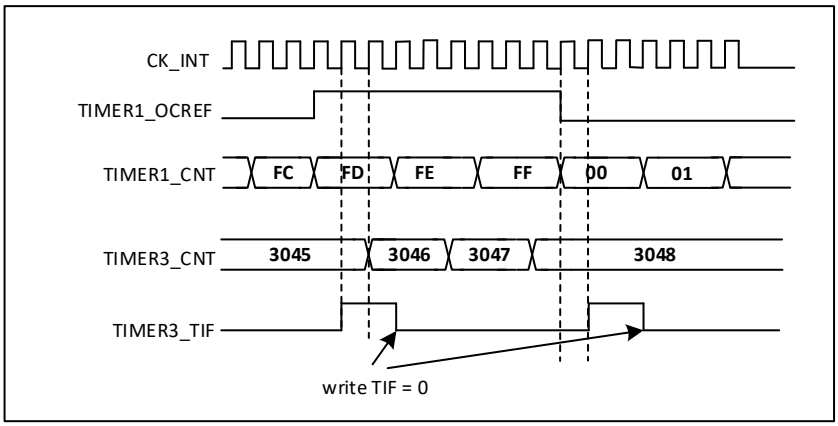

Figure 18-39 Gating timer 3 with OC1REF of timer 1

In the example in figure xx, the Timer 3 counter and prescaler are not initialized before being started. So they start counting from their current value. It is possible to start from a given value by resetting both timers before starting Timer 1. Then any value can be written in the timer counters. The timers can easily be reset by software using the UG bit in the TIMx_EGR registers.

In the next example, we synchronize Timer 1 and Timer 3. Timer 1 is the master and starts from 0. Timer 3 is the slave and starts from 0xE7. The prescaler ratio is the same for both timers. Timer 3 stops when Timer 1 is disabled by writing ‘0 to the CEN bit in the TIM1_CR1 register:

- Configure Timer 1 master mode to send its Counter Enable signal (CNT_EN) as a trigger output (MMS = 001 in the TIM1_CR2 register).
- Configure the Timer 1 OC1REF waveform (TIM1_CCMR1 register).
- Configure Timer 3 to get the input trigger from Timer 1 (TS = 000 in the TIM3_SMCR register).
- Configure Timer 3 in gated mode (SMS = 101 in TIM3_SMCR register).
- Reset Timer 1 by writing ‘1 in UG bit (TIM1_EGR register).
- Reset Timer 3 by writing ‘1 in UG bit (TIM3_EGR register).
- Initialize Timer 3 to 0xE7 by writing ‘0xE7’ in the timer 3 counter (TIM3_CNT).
- Enable Timer 3 by writing ‘1 in the CEN bit (TIM3_CR1 register).
- Start Timer 1 by writing ‘1 in the CEN bit (TIM1_CR1 register).
- Stop Timer 1 by writing ‘0 in the CEN bit (TIM1_CR1 register).

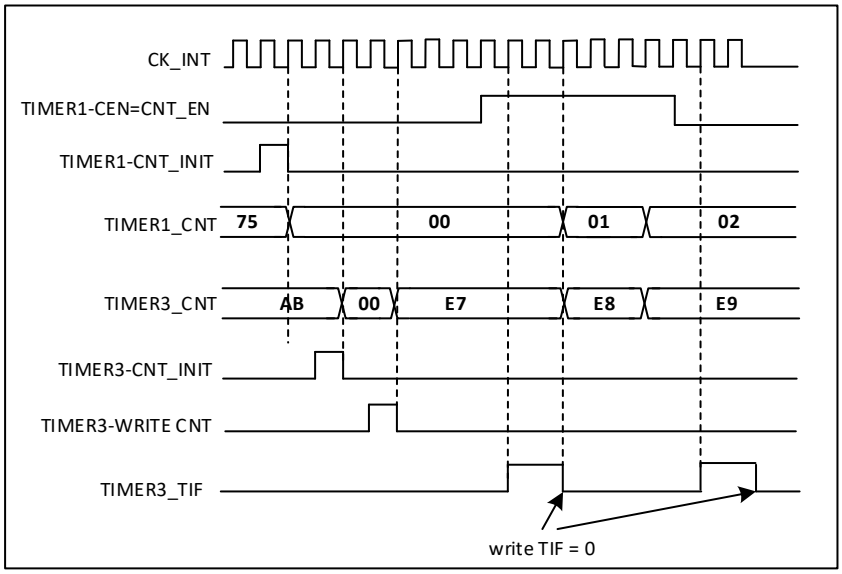

Figure 18-40 Gating timer 3 with Enable of timer 1

Using one timer to start another timer

In this example, we set the enable of Timer 3 with the update event of Timer 1. Refer to Figure 132 for connections. Timer 3 starts counting from its current value (which can be nonzero) on the divided internal clock as soon as the update event is generated by Timer 1. When Timer 3 receives the trigger signal its CEN bit is automatically set and the counter counts until we write '0' to the CEN bit in the TIM3_CR1 register. Both counter clock frequencies are divided by 3 by the prescaler compared to CK_INT ($f_{CK\_CNT} = f_{CK\_INT}/3$).

- Configure Timer 1 master mode to send its Update Event (UEV) as trigger output (MMS = 010 in the TIM1_CR2 register).
- Configure the Timer 1 period (TIM1_ARR registers).
- Configure Timer 3 to get the input trigger from Timer 1 (TS = 000 in the TIM3_SMCR register).
- Configure Timer 3 in trigger mode (SMS = 110 in TIM3_SMCR register).
- Start Timer 1 by writing '1 in the CEN bit (TIM1_CR1 register).

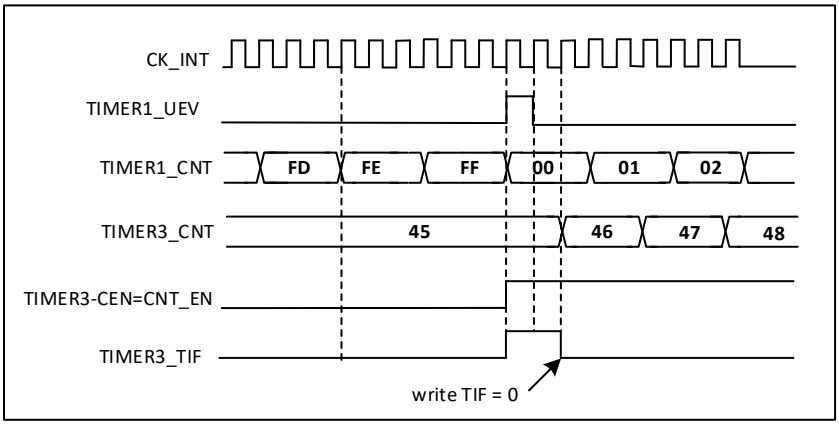

Figure 18-41 Triggering timer 3 with update of timer 1

As in the previous example, both counters can be initialized before starting counting. The following Figure shows the behavior with the same configuration as in the previous Figure but in trigger mode instead of gated mode (SMS = 110 in the TIM3_SMCR register).

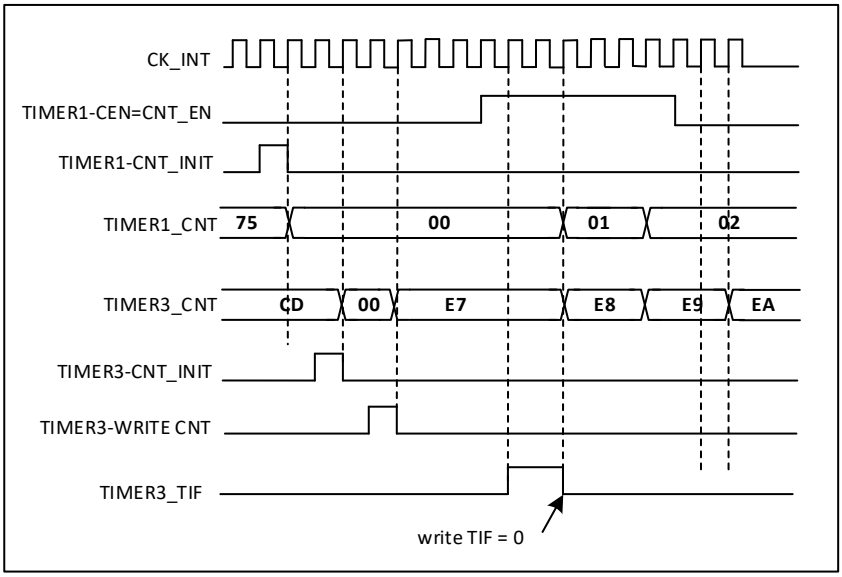

Figure 18-42 Triggering timer 3 with Enable of timer 1

Use one timer as a prescaler for another

In this example, we set the enable of timer 1 when its TI1 input rises, and the enable of Timer 3 with the enable of Timer 1. Refer to Figure 132 for connections. To ensure the counters are aligned, Timer 1 must be configured in Master/Slave mode (slave with respect to TI1, master with respect to Timer 3):

- Configure Timer 1 master mode to send its Enable as trigger output (MMS = 001 in the TIM1_CR2 register).
- Configure Timer 1 slave mode to get the input trigger from TI1 (TS = 100 in the TIM1_SMCR register).
- Configure Timer 1 in trigger mode (SMS = 110 in the TIM1_SMCR register).
- Configure the Timer 1 in Master/Slave mode by writing MSM = 1 (TIM1_SMCR register).
- Configure Timer 3 to get the input trigger from Timer 1 (TS = 000 in the TIM3_SMCR register).
- Configure Timer 3 in trigger mode (SMS = 110 in the TIM3_SMCR register).

When a rising edge occurs on TI1 (Timer 1), both counters starts counting synchronously on the internal clock and both TIF flags are set.

Note: In this example both timers are initialized before starting (by setting their respective UG bits). Both counters starts from 0, but an offset can easily be inserted between them by writing any of the counter registers (TIMx_CNT). One can see that the master/slave mode insert a delay between CNT_EN and CK_PSC on timer1.

18.3.15. Debug mode

When the microcontroller enters debug mode, the TIM3 counter either continues to work normally or stops, depending on DBG_TIMx_STOP configuration bit in DBGMCU module.

18.4. TIM3 registers

18.4.1. TIM3 control register 1 (TIM3_CR1)

Address offset: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res

-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	CKD[1:0]		ARPE	CMS[1:0]		DIR	OPM	URS	UDIS	CEN
-	-	-	-	-	-	RW		RW	RW		RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved			
9:8	CKD[1:0]	RW	00	Clock division This bit-field indicates the division ratio between the timer clock (CK_INT) frequency and sampling clock used by the digital filters (ETR, TIx) 00: tDTS = tCK_INT 01: tDTS = 2 x tCK_INT 10: tDTS = 4 x tCK_INT 11: Reserved
7	ARPE	RW	0	Auto-reload preload enable 0: TIM3_ARR register is not buffered 1: TIM3_ARR register is buffered
6:5	CMS[1:0]	RW	00	Center-aligned mode selection 00: Edge-aligned mode. The counter counts up or down depending on the direction bit(DIR). 01: Center-aligned mode 1. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS = 00 in TIM3_CCMRx register) are set only when the counter is counting down. 10: Center-aligned mode 2. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS = 00 in TIM3_CCMRx register) are set only when the counter is counting up. 11: Center-aligned mode 3. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS = 00 in TIM3_CCMRx register) are set both when the counter is counting up or down. Note: It is not allowed to switch from edge-aligned mode to center-aligned mode as long as the counter is enabled (CEN = 1)
4	DIR	RW	0	Direction 0: Counter used as upcounter 1: Counter used as downcounter Note: This bit is read only when the timer is configured in Center-aligned mode or Encoder mode.
3	OPM	RW	0	One-pulse mode 0: Counter is not stopped at update event 1: Counter stops counting at the next update event (clearing the bit CEN)
2	URS	RW	0	Update request source This bit is set and cleared by software to select the UEV event sources. 0: Any of the following events generate an update interrupt or DMA request if enabled. These events can be: − Counter overflow/underflow − Setting the UG bit − Update generation through the slave mode controller 1: Only counter overflow/underflow generates an update interrupt or DMA request if enabled.
1	UDIS	RW	0	Update disable This bit is set and cleared by software to enable/disable UEV event generation. 0: UEV enabled. The Update (UEV) event is generated by one of the following events: − Counter overflow/underflow − Setting the UG bit − Update generation through the slave mode controller Buffered registers are then loaded with their preload values.

				1: UEV disabled. The Update event is not generated, shadow registers keep their value (ARR, PSC, CCRx). However the counter and the prescaler are reinitialized if the UG bit is set or if a hardware reset is received from the slave mode controller.
0	CEN	RW	0	Counter enable 0: Counter disabled 1: Counter enabled Note: External clock, gated mode and encoder mode can work only if the CEN bit has been previously set by software. However trigger mode can set the CEN bit automatically by hardware.

18.4.2. TIM3 control register 2 (TIM3_CR2)

Address offset: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	TI1S	MMS[2:0]			CCDS	Res	Res	Res
-	-	-	-	-	-	-	-	RW	RW	RW	RW	RW	-	-	-

Bit	Name	R/W	Reset Value	Function
31: 8	Reserved	-	0	Reserved, must be kept at reset value.
7	TI1S	RW	0	TI1 selection 0: The TIM3_CH1 pin is connected to TI1 input 1: The TIM3_CH1, CH2 and CH3 pins are connected to the TI1 input (XOR combination)
6:4	MMS[2:0]	RW	000	Master mode selection These bits allow to select the information to be sent in master mode to slave timers for synchronization (TRGO). The combination is as follows: 000: Reset - the UG bit from the TIMx_EGR register is used as trigger output (TRGO). If the reset is generated by the trigger input (slave mode controller configured in reset mode) then the signal on TRGO is delayed compared to the actual reset. 001: Enable - the Counter enable signal, CNT_EN, is used as trigger output (TRGO). It is useful to start several timers at the same time or to control a window in which a slave timer is enabled. The Counter Enable signal is generated by a logic OR between CEN control bit and the trigger input when configured in gated mode. When the Counter Enable signal is controlled by the trigger input, there is a delay on TRGO, except if the master/slave mode is selected (see the MSM bit description in TIM3_SMCR register). 010: Update - The update event is selected as trigger output (TRGO). For instance a master timer can then be used as a prescaler for a slave timer. 011: Compare Pulse - The trigger output send a positive pulse when the CC1IF flag is to be set (even if it was already high), as soon as a capture or a compare match occurred.(TRGO) 100: Compare - OC1REF signal is used as trigger output (TRGO) 101: Compare - OC2REF signal is used as trigger output (TRGO) 110: Compare - OC3REF signal is used as trigger output (TRGO) 111: Compare - OC4REF signal is used as trigger output (TRGO)
3	CCDS	RW	0	Capture/compare DMA selection 0: CCx DMA request sent when CCx event occurs 1: CCx DMA requests sent when update event occurs

2:0	Reseved	-	0	Reserved, must be kept at reset value.

18.4.3. TIM3 slave mode control register (TIM3_SMCR)

Address offset: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res		Res				MSM	TS[2:0]			OCCS	SMS[2:0]		
								RW	RW			RW	RW		

Bit	Name	R/W	Reset Value	Function
31:8	Reserved			
7	MSM			Master/Slave mode 0: No action 1: The effect of an event on the trigger input (TRGI) is delayed to allow a perfect synchronization between the current timer and its slaves (through TRGO). It is useful if we want to synchronize several timers on a single external event.
6:4	TS[2:0]			Trigger selection This bit-field selects the trigger input to be used to synchronize the counter. 000: Internal Trigger 0 (ITR0). 001: Internal Trigger 1 (ITR1). 010: Internal Trigger 2 (ITR2). 011: Internal Trigger 3 (ITR3). 100: TI1 Edge Detector (TI1F_ED) 101: Filtered Timer Input 1 (TI1FP1) 110: Filtered Timer Input 2 (TI2FP2) 111: Reserved Note: These bits must be changed only when they are not used to avoid wrong edge detections at the transition.
3	OCCS			OCREF clear selection. This bit is used to select the OCREF clear source. 0:OCREF_CLR_INT is connected to the OCREF_CLR input 1: OCREF_CLR_INT is connected to ETRF
2:0	SMS[2:0]			Slave mode selection When external signals are selected the active edge of the trigger signal (TRGI) is linked to thepolarity selected on the external input (see Input Control register and Control Register description. 000: Slave mode disabled - if CEN = ‘1 then the prescaler is clocked directly by the internal clock. 001: Encoder mode 1 - Counter counts up/down on TI2FP2 edge depending on TI1FP1 level. 010: Encoder mode 2 - Counter counts up/down on TI1FP1 edge depending on TI2FP2 level. 011: Encoder mode 3 - Counter counts up/down on both TI1FP1 and TI2FP2 edges depending on the level of the other input. 100: Reset Mode - Rising edge of the selected trigger input (TRGI) reinitializes the counter and generates an update of the registers. 101: Gated Mode - The counter clock is enabled when the trigger input (TRGI) is high. The counter stops (but is not reset) as soon as the trigger becomes low. Both start and stop of the counter are controlled. 110: Trigger Mode - The counter starts at a rising edge of the trigger TRGI (but it is not reset). Only the start of the counter is controlled. 111: External Clock Mode 1 - Rising edges of the selected trigger (TRGI) clock the counter.

				Note: The gated mode must not be used if TI1F_ED is selected as the trigger input (TS = 100). Indeed, TI1F_ED outputs 1 pulse for each transition on TI1F, whereas the gated mode checks the level of the trigger signal.

Table 18-2 TIM3 internal trigger connection

Slave TIM	ITR0(TS = 000)	ITR1(TS = 001)	ITR2(TS = 010)	ITR3(TS = 011)
TIM3	TIM1	Reserved	Reserved	TIM14 OC1

18.4.4. TIM3 DMA/Interrupt enable register (TIM3_DIER)

Address offset: 0x0C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	TDE	Res	CC4DE	CC3DE	CC2DE	CC1DE	UDE	Res	TIE	Res	CC4IE	CC3IE	CC2IE	CC1IE	UIE
-	RW	-	RW	RW	RW	RW	RW	-	RW	-	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:15	Reserved			Reserved, must be kept at reset value.
14	TDE	RW	0	TDE: Trigger DMA request enable 0: Trigger DMA request disabled. 1: Trigger DMA request enabled.
13	Reserved	-	0	Reserved, must be kept at reset value.
12	CC4DE	RW	0	CC4DE: Capture/Compare 4 DMA request enable 0: CC4 DMA request disabled. 1: CC4 DMA request enabled.
11	CC3DE	RW	0	CC3DE: Capture/Compare 3 DMA request enable 0: CC3 DMA request disabled. 1: CC3 DMA request enabled.
10	CC2DE	RW	0	CC2DE: Capture/Compare 2 DMA request enable 0: CC2 DMA request disabled. 1: CC2 DMA request enabled.
9	CC1DE	RW	0	CC1DE: Capture/Compare 1 DMA request enable 0: CC1 DMA request disabled. 1: CC1 DMA request enabled.
8	UDE	RW	0	UDE: Update DMA request enable 0: Update DMA request disabled. 1: Update DMA request enabled.
7	Reserved	-	0	Reserved, must be kept at reset value.
6	TIE	RW	0	TIE: Trigger interrupt enable 0: Trigger interrupt disabled. 1: Trigger interrupt enabled.
5	Reserved	-	0	Reserved, must be kept at reset value.
4	CC4IE	RW	0	CC4IE: Capture/Compare 4 interrupt enable 0: CC4 interrupt disabled. 1: CC4 interrupt enabled.
3	CC3IE	RW	0	CC3IE: Capture/Compare 3 interrupt enable 0: CC3 interrupt disabled 1: CC3 interrupt enabled
2	CC2IE	RW	0	CC2IE: Capture/Compare 2 interrupt enable 0: CC2 interrupt disabled 1: CC2 interrupt enabled
1	CC1IE	RW	0	CC1IE: Capture/Compare 1 interrupt enable 0: CC1 interrupt disabled 1: CC1 interrupt enabled
0	UIE	RW	0	UIE: Update interrupt enable 0: Update interrupt disabled 1: Update interrupt enabled

18.4.5. TIM3 status register (TIM3_SR)

Address offset: 0x010

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	CC4OF	CC3OF	CC2OF	CC1OF	Res	Res	TIF	Res	CC4IF	CC3IF	CC2IF	CC1F	UIF
-	-	-	Rc_w0	Rc_w0	Rc_w0	Rc_w0	-	-	Rc_w0	-	Rc_w0	Rc_w0	Rc_w0	Rc_w0	Rc_w0

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	0	Reserved, must be kept at reset value.
12	CC4OF	Rc_w0	0	Capture/Compare 4 overcapture flag Refer to CC1OF description
11	CC3OF	Rc_w0	0	Capture/Compare 3 overcapture flag Refer to CC1OF description
10	CC2OF	Rc_w0	0	Capture/compare 2 overcapture flag Refer to CC1OF description
9	CC1F	Rc_w0	0	This flag is set by hardware only when the corresponding channel is configured in input capture mode. It is cleared by software by writing it to 0. 0: No overcapture has been detected 1: The counter value has been captured in TIM3_CCR1 register while CC1IF flag was already set
8:7	Res	-	0	Reserved, must be kept at reset value.
6	TIF	Rc_w0	0	Trigger interrupt flag This flag is set by hardware on trigger event (active edge detected on TRGI input when the slave mode controller is enabled in all modes but gated mode. It is set when the counter starts or stops when gated mode is selected). It is cleared by software. 0: No trigger event occurred 1: Trigger interrupt pending
5	Res	-	0	Reserved, must be kept at reset value.
4	CC4IF	Rc_w0	0	Capture/Compare 4 interrupt flag Refer to CC1IF description
3	CC3IF	Rc_w0	0	Capture/Compare 3 interrupt flag Refer to CC1IF description
2	CC2IF	Rc_w0	0	Capture/Compare 2 interrupt flag Refer to CC1IF description
1	CC1IF	Rc_w0	0	Capture/compare 1 interrupt flag If channel CC1 is configured as output: This flag is set by hardware when the counter matches the compare value, with some exception in center-aligned mode (refer to the CMS bits in the TIMx_CR1 register description). It is cleared by software. 0: No match 1: The content of the counter TIM3_CNT matches the content of the TIM3_CCR1 register. If channel CC1 is configured as input: This bit is set by hardware on a capture. It is cleared by software or by reading the TIM3_CCR1 register. 0: No input capture occurred 1: The counter value has been captured in TIM3_CCR1 register (An edge has been detected on IC1 which matches the selected polarity)
0	UIF	Rc_w0	0	Update interrupt flag This bit is set by hardware on an update event. It is cleared by software. 0: No update occurred. 1: Update interrupt pending. This bit is set by hardware when the registers are updated:

				− At overflow or underflow and if UDIS = 0 in the TIM3_CR1 register. − When CNT is reinitialized by software using the UG bit in TIM3_EGR register, if URS = 0 andUDIS = 0 in the TIM3_CR1 register. − When CNT is reinitialized by a trigger event (refer to the slave mode control register(TIM3_SMCR) description), if URS = 0 and UDIS = 0 in the TIM3_CR1 register.

18.4.6. TIM3 event generation register (TIM3_EGR)

Address offset: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	Res	TG	Res	CC4G	CC3G	CC2G	CC1G	UG
-	-	-	--	-	-	-	-	-	W	-	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	0	Reserved, must be kept at reset value.
7	Reserved	-	0	Reserved, must be kept at reset value.
6	TG	W	0	Trigger generation This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: The TIF flag is set in TIMx_SR register. Related interrupt or DMA transfer can occur if enabled.
5	Reserved	-	0	Reserved, must be kept at reset value.
4	CC4G	W	0	Capture/compare 4 generation Refer to CC1G description
3	CC3G	W	0	Capture/compare 3 generation Refer to CC1G description
2	CC2G	W	0	Capture/compare 2 generation Refer to CC1G description
1	CC1G	W	0	Capture/compare 1 generation This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: A capture/compare event is generated on channel 1: If channel CC1 is configured as output: CC1IF flag is set, Corresponding interrupt or DMA request is sent if enabled. If channel CC1 is configured as input: The current value of the counter is captured in TIMx_CCR1 register. The CC1IF flag is set, the corresponding interrupt or DMA request is sent if enabled. The CC1OF flag is set if the CC1IF flag was already high.
0	UG	W	0	Update generation This bit can be set by software, it is automatically cleared by hardware. 0: No action 1: Re-initialize the counter and generates an update of the registers. Note that the prescaler counter is cleared too (anyway the prescaler ratio is not affected). The counter is cleared if the center-aligned mode is selected or if DIR = 0 (upcounting), else it takes the auto-reload value (TIM3_ARR) if DIR = 1 (downcounting).

18.4.7. TIM3 capture/compare mode register 1 (TIM3_CCMR1)

Address offset: 0x18

Reset value: 0x0000 0000

Output compare mode:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
OC2CE	OC2M[2:0]			OC2PE	CO2FE	CC2S[1:0]		OC1CE	OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	
IC2F[3:0]				IC2PSC[1:0]				IC1F[3:0]				IC1PSC[1:0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Output compare mode

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-		Reserved, must be kept at reset value.
15	OC2CE	RW	0	Output compare 2 clear enable
14:12	OC2M[2:0]	RW	000	Output compare 2 mode
11	OC2PE	RW	0	Output compare 2 preload enable
10	OC2FE	RW	0	Output compare 2 fast enable
9:8	CC2S[1:0]	RW	00	Capture/Compare 2 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC2 channel is configured as output 01: CC2 channel is configured as input, IC2 is mapped on TI2 10: CC2 channel is configured as input, IC2 is mapped on TI1 11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode is working only if an internal trigger input is selected through the TS bit (TIMx_SMCR register) Note: CC2S bits are writable only when the channel is OFF (CC2E = 0 in TIM3_CCER).
7	OC1CE	RW	0	Output Compare 1 Clear Enable 0: OC1REF is not affected by the ETRF input 1: OC1REF is cleared as soon as a High level is detected on ETRF input
6:4	OC1M[2:0]	RW	00	Output compare 1 mode These bits define the behavior of the output reference signal OC1REF from which OC1 and OC1N are derived. OC1REF is active high whereas OC1 and OC1N active level depends on CC1P and CC1NP bits. 000: Frozen - The comparison between the output compare register TIM3_CCR1 and the counter TIM3_CNT has no effect on the outputs. 001: Set channel 1 to active level on match. OC1REF signal is forced high when the counter TIMx_CNT matches the capture/compare register 1 (TIMx_CCR1). 010: Set channel 1 to inactive level on match. OC1REF signal is forced low when the counter TIMx_CNT matches the capture/compare register 1 (TIMx_CCR1).011: Toggle - OC1REF toggles when TIM3_CNT = TIM3_CCR1. 100: Force inactive level - OC1REF is forced low. 101: Force active level - OC1REF is forced high. 110: PWM mode 1 - In upcounting, channel 1 is active as long as TIMx_CNT < TIMx_CCR1 else inactive. In downcounting, channel 1 is inactive (OC1REF = ‘0) as long as TIMx_CNT > TIMx_CCR1 else active (OC1REF = 1). 111: PWM mode 2 - In upcounting, channel 1 is inactive as long as TIMx_CNT < TIMx_CCR1 else active. In downcounting, channel 1 is active as long as TIMx_CNT > TIMx_CCR1 else inactive. Note: 1: These bits can not be modified as long as LOCK level 3 has been programmed (LOCK bits in TIMx_BDTR register) and CC1S = 00 (the channel is configured in output). 2: In PWM mode 1 or 2, the OCREF level changes only when the result of the comparison changes or when the output compare mode switches from “frozen” mode to “PWM” mode.
3	OC1PE	RW	0	Output compare 1 preload enable

				0: Preload register on TIM3_CCR1 disabled. TIM3_CCR1 can be written at anytime, the new value is taken in account immediately. 1: Preload register on TIM3_CCR1 enabled. Read/Write operations access the preload register. TIM3_CCR1 preload value is loaded in the active register at each update event. Note: 1: These bits can not be modified as long as LOCK level 3 has been programmed (LOCK bits in TIMx_BDTR register) and CC1S = 00 (the channel is configured in output). 2: Only in one pulse mode, PWM mode can be used without confirming the preload register, otheRWise its behavior is undefined.
2	OC1FE	RW	0	Output compare 1 fast enable This bit is used to accelerate the effect of an event on the trigger in input on the CC output. 0: CC1 behaves normally depending on counter and CCR1 values even when the trigger is ON. The minimum delay to activate CC1 output when an edge occurs on the trigger input is 5 clock cycles. 1: An active edge on the trigger input acts like a compare match on CC1 output. Then, OC is set to the compare level independently from the result of the comparison. Delay to sample the trigger input and to activate CC1 output is reduced to 3 clock cycles. OCFE acts only if the channel is configured in PWM1 or PWM2 mode.
1:0	CC1S[1:0]	RW	00	Capture/Compare 1 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC1 channel is configured as output. 01: CC1 channel is configured as input, IC1 is mapped on TI1. 10: CC1 channel is configured as input, IC1 is mapped on TI2. 11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIMx_SMCR register) Note: CC1S bits are writable only when the channel is OFF (CC1E = 0 in TIM3_CCER).

Input Capture mode:

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-		Reserved, must be kept at reset value.
15:12	IF2F	RW	0000	Input capture 2 filter
11:10	IC2PSC[1:0]	RW	00	Input capture 2 prescaler
9:8	CC2S[1:0]	RW	0	Capture/compare 2 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC2 channel is configured as output. 01: CC2 channel is configured as input, IC2 is mapped on TI2. 10: CC2 channel is configured as input, IC2 is mapped on TI1. 11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIM3_SMCR register) Note: CC2S bits are writable only when the channel is OFF (CC2E = 0 in TIM3_CCER).
7:4	IC1F[3:0]	RW	0000	Input capture 1 filter This bit-field defines the frequency used to sample TI1 input and the length of the digital filter applied to TI1. The digital filter is made of an event counter in which N consecutive events are needed to validate a transition on the output: 0000: No filter, sampling is done at fDTS 0001: fSAMPLING = fCK_INT, N = 2 0010: fSAMPLING = fCK_INT, N = 4 0011: fSAMPLING = fCK_INT, N = 8 0100: fSAMPLING = fDTS / 2, N = 6

				0101: fSAMPLING = fDTS / 2, N = 8 0110: fSAMPLING = fDTS / 4, N = 6 0111: fSAMPLING = fDTS / 4, N = 8 1000: fSAMPLING = fDTS / 8, N = 6 1001: fSAMPLING = fDTS / 8, N = 8 1010: fSAMPLING = fDTS / 16, N = 5 1011: fSAMPLING = fDTS / 16, N = 6 1100: fSAMPLING = fDTS / 16, N = 8 1101: fSAMPLING = fDTS / 32, N = 5 1110: fSAMPLING = fDTS / 32, N = 6 1111: fSAMPLING = fDTS / 32, N = 8
3:2	IC1PSC[1:0]	RW	00	Input capture 1 prescaler This bit-field defines the ratio of the prescaler acting on CC1 input (IC1). The prescaler is reset as soon as CC1E = 0 (TIM1_CCER register). 00: no prescaler, capture is done each time an edge is detected on the capture input 01: capture is done once every 2 events 10: capture is done once every 4 events 11: capture is done once every 8 events
1:0	CC1S[1:0]	RW	00	Capture/Compare 1 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC1 channel is configured as output 01: CC1 channel is configured as input, IC1 is mapped on TI1 10: CC1 channel is configured as input, IC1 is mapped on TI2 11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIM3_SMCR register) Note: CC1S bits are writable only when the channel is OFF (CC1E = 0 in TIMx_CCER).

18.4.8. TIM3 capture/compare mode register 2 (TIM3_CCMR2)

Address offset: 0x1C

Reset value: 0x0000 0000

Output compare mode:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
OC4CE	OC4M[2:0]			OC4PE	CO4FE	CC4S[1:0]		OC3CE	OC3M[2:0]			OC3PE	OC3FE	CC3S[1:0]	
IC4F[3:0]				IC4PSC[1:0]				IC3F[3:0]				IC3PSC[1:0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Output compare mode

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-		Reserved, must be kept at reset value.
15	OC4CE	RW	0	Output compare 4 clear enable
14:12	OC4M[2:0]	RW	000	Output compare 4 mode
11	OC4PE	RW	0	Output compare 4 preload enable
10	OC4FE	RW	0	Output compare 4 fast enable
9:8	CC4S[1:0]	RW	00	Capture/Compare 4 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC4 channel is configured as output 01: CC4 channel is configured as input, IC4 is mapped on TI4 10: CC4 channel is configured as input, IC4 is mapped on TI3

				11: CC4 channel is configured as input, IC4 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIMx_SMCR register) Note: CC4S bits are writable only when the channel is OFF (CC4E = 0 in TIM1_CCER).
7	OC3CE	RW	0	Output compare 3 clear enable
6:4	OC3M[2:0]	RW	00	Output compare 3 mode
3	OC3PE	RW	0	Output compare 3 preload enable
2	OC3FE	RW	0	Output compare 3 fast enable
1:0	CC3S[1:0]	RW	00	Capture/Compare 3 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC3 channel is configured as output 01: CC3 channel is configured as input, IC3 is mapped on TI3 10: CC3 channel is configured as input, IC3 is mapped on TI4 11: CC3 channel is configured as input, IC3 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIMx_SMCR register) Note: CC3S bits are writable only when the channel is OFF (CC3E = 0 in TIMx_CCER).

Input Capture mode:

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-		Reserved, must be kept at reset value.
15:12	IF4F	RW	0000	Input capture 4 filter
11:10	IC4PSC[1:0]	RW	00	Input capture 4 prescaler
9:8	CC4S[1:0]	RW	0	Capture/Compare 4 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC4 channel is configured as output 01: CC4 channel is configured as input, IC4 is mapped on TI4 10: CC4 channel is configured as input, IC4 is mapped on TI3 11: CC4 channel is configured as input, IC4 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIM3_SMCR register) Note: CC4S bits are writable only when the channel is OFF (CC4E = 0 in TIMx_CCER).
7:4	IC3F[3:0]	RW	0000	Input capture 3 filter This bit-field defines the frequency used to sample TI3 input and the length of the digital filter applied to TI3. The digital filter is made of an event counter in which N consecutive events are needed to validate a transition on the output: 0000: No filter, sampling is done at fDTS 0001: fSAMPLING = fCK_INT, N = 2 0010: fSAMPLING = fCK_INT, N = 4 0011: fSAMPLING = fCK_INT, N = 8 0100: fSAMPLING = fDTS / 2, N = 6 0101: fSAMPLING = fDTS / 2, N = 8 0110: fSAMPLING = fDTS / 4, N = 6 0111: fSAMPLING = fDTS / 4, N = 8 1000: fSAMPLING = fDTS / 8, N = 6 1001: fSAMPLING = fDTS / 8, N = 8 1010: fSAMPLING = fDTS / 16, N = 5 1011: fSAMPLING = fDTS / 16, N = 6 1100: fSAMPLING = fDTS / 16, N = 8 1101: fSAMPLING = fDTS / 32, N = 5 1110: fSAMPLING = fDTS / 32, N = 6 1111: fSAMPLING = fDTS / 32, N = 8
3:2	IC3PSC[1:0]	RW	00	Input capture 3 prescaler This bit-field defines the ratio of the prescaler acting on CC3 input (IC1). The prescaler is reset as soon as CC3E = 0 (TIMx_CCER register). 00: no prescaler, capture is done each time an edge is detected on the capture input

				01: capture is done once every 2 events 10: capture is done once every 4 events 11: capture is done once every 8 events
1:0	CC3S[1:0]	RW	00	Capture/Compare 3 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC3 channel is configured as output 01: CC3 channel is configured as input, IC3 is mapped on TI3 10: CC3 channel is configured as input, IC3 is mapped on TI4 11: CC3 channel is configured as input, IC3 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIM3_SMCR register) Note: CC3S bits are writable only when the channel is OFF (CC3E = 0 in TIM3_CCER).

18.4.9. TIM3 capture/compare enable register (TIM3_CCER)

Address offset: 0x20

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CC4NP	Res	CC4P	CC4E	CC3NP	Res	CC3P	CC3E	CC2NP	Res	CC2P	CC2E	CC1NP	Res	CC1P	CC1E
RW	-	RW	RW	RW	-	RW	RW	RW	-	RW	RW	RW	-	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	0	Reserved, must be kept at reset value.
15	CCNP	RW	0	Capture/Compare 4 output Polarity. Refer to CC1NP description
14	Reserved	-	0	Reserved, must be kept at reset value.
13	CC4P	RW	0	Capture/Compare 4 output Polarity. Refer to CC1P description
12	Reserved	-	0	Capture/Compare 4 output enable. Refer to CC1E description
11	CC3NP	RW	0	Capture/Compare 3 output Polarity. Refer to CC1NP description
10	Reserved	-	0	Reserved, must be kept at reset value.
9	CC3P	RW	0	Capture/Compare 3 output Polarity. Refer to CC1P description
8	CC3E	RW	0	Capture/Compare 3 output enable. Refer to CC1E description
7	CC2NP	RW	0	Capture/Compare 2 output Polarity. Refer to CC1NP description
6	Reserved	-	0	Reserved, must be kept at reset value.
5	CC2P	RW	0	Capture/Compare 2 output Polarity. Refer to CC1P description
4	CC2E	RW	0	Capture/Compare 2 output enable. Refer to CC1E description
3	CC1NP	RW	0	Capture/Compare 1 output Polarity 0: OC1N active high 1: OC1N active low This bit is used in conjunction with CC1P to define TI1FP1/TI2FP1 polarity. refer to CC1P description.
2	Reserved	-	0	Reserved, must be kept at reset value.
1	CC1P	RW	0	Capture/Compare 1 output Polarity. CC1 channel configured as output: 0: OC1 active high 1: OC1 active low CC1 channel configured as input: CC1NP/CC1P bits select TI1FP1 and TI2FP1 polarity for trigger or capture operations. 00: noninverted/rising edge

				Circuit is sensitive to TIxFP1 rising edge (capture, trigger in reset, external clock or trigger mode), TIxFP1 is not inverted (trigger in gated mode, encoder mode). 01: inverted/falling edge Circuit is sensitive to TIxFP1 falling edge (capture, trigger in reset, external clock or trigger mode), TIxFP1 is inverted (trigger in gated mode, encoder mode). 10: reserved, do not use this configuration. 11: noninverted/both edges
0	CC1E	RW	0	Capture/Compare 1 output enable. CC1 channel configured as output: 0: Off - OC1 is not active 1: On - OC1 signal is output on the corresponding output pin CC1 channel configured as input: This bit determines if a capture of the counter value can actually be done into the input capture/compare register 1 (TIMx_CCR1) or not. 0: Capture disabled 1: Capture enabled

CcxE bit	OCx output State
0	Output disabled (OCx = 0,OCx_EN = 0)
1	OCx = OCxREF+Polarity,OCx_EN = 1

18.4.10. TIM3 counter (TIM3_CNT)

Address offset: 0x24

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CNT[15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			Reserved, must be kept at reset value.
15:0	CNT[15:0]	RW	0	counter value

18.4.11. TIM3 prescaler (TIM3_PSC)

Address offset: 0x28

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PSC[15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			Reserved, must be kept at reset value.
15:0	PSC[15:0]	RW	0	Prescaler value The counter clock frequency CK_CNT is equal to fCK_PSC / (PSC[15:0] + 1). PSC contains the value to be loaded in the active prescaler register at each update event.

18.4.12. TIM3 auto-reload register (TIM3_ARR)

Address offset: 0x2C

Reset value: 0x0000 FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ARR[15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			Reserved, must be kept at reset value.
15:0	ARR[15:0]	RW	0	Auto-reload value ARR is the value to be loaded in the actual auto-reload register. Refer to Section 12.4.1: Time-base unit for more details about ARR update and behavior. The counter is blocked while the auto-reload value is null.

18.4.13. TIM3 capture/compare register 1 (TIM3_CCR1)

Address offset: 0x34

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR1[15:0]															
RW/RO															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			Reserved, must be kept at reset value.
15:0	CCR1[15:0]	RW	0	Capture/Compare 1 value If channel CC1 is configured as output: CCR1 is the value to be loaded in the actual capture/compare 1 register (preload value). It is loaded permanently if the preload feature is not selected in the TIM3_CCMR1 register (bit OC1PE). OtheRWise the preload value is copied in the active capture/compare 1 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIM3_CNT and signaled on OC1 output. If channel CC1is configured as input: CCR1 is the counter value transferred by the last input capture 1 event (IC1).

18.4.14. TIM3 capture/compare register 2 (TIM3_CCR2)

Address offset: 0x38

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR2[15:0]															
RW/RO															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			Reserved, must be kept at reset value.
15:0	CCR2[15:0]	RW	0	Capture/Compare 2 value If channel CC2 is configured as output: CCR2 is the value to be loaded in the actual capture/compare 2 register (preload value).

				It is loaded permanently if the preload feature is not selected in the TIM3_CCMR2 register (bit OC2PE). Else the preload value is copied in the active capture/compare 2 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIM3_CNT and signaled on OC2 output. If channel CC2 is configured as input: CCR2 is the counter value transferred by the last input capture 2 event (IC2).

18.4.15. TIM3 capture/compare register 3 (TIM3_CCR3)

Address offset: 0x3C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR3[15:0]															
RW/RO															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			Reserved, must be kept at reset value.
15:0	CCR3[15:0]	RW	0	Capture/Compare 3 value If channel CC3 is configured as output: CCR3 is the value to be loaded in the actual capture/compare 3 register (preload value). It is loaded permanently if the preload feature is not selected in the TIM3_CCMR3 register (bit OC3PE). Else the preload value is copied in the active capture/compare 3 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIM3_CNT and signaled on OC3 output. If channel CC3is configured as input: CCR3 is the counter value transferred by the last input capture 3 event (IC3).

18.4.16. TIM3 capture/compare register 4 (TIM3_CCR4)

Address offset: 0x40

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR4[15:0]															
RW/RO															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			Reserved, must be kept at reset value.
15:0	CCR4[15:0]	RW	0	Capture/Compare 4 value If CC4 channel is configured as output: CCR4 is the value to be loaded in the actual capture/compare 4 register (preload value). It is loaded permanently if the preload feature is not selected in the TIMx_CCMR4 register (bit OC4PE). OtheRWise, the preload value is copied in the active capture/compare 4 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC4 output. If CC4 channel is configured as input:

				CCR4 is the counter value transferred by the last input capture 4 event (IC4).

18.4.17. TIM3 DMA control register (TIM3_DCR)

Address offset: 0x48

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	DBL[4:0]					Res			DBA[4:0]				
-	-	-	RW	RW	RW	RW	RW	-	-	-	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:13	Reserved			Reserved, must be kept at reset value.
12:8	DBL[4:0]	RW	0 0000	DMA burst length This 5-bit vector defines the number of DMA transfers (the timer recognizes a burst transfer when a read or a write access is done to the TIMx_DMAR address). 00000: 1 transfer, 00001: 2 transfers, 00010: 3 transfers, ... 10001: 18 transfers.
7:5	Reserved	RW	0	Reserved, must be kept at reset value.
4:0	DBA[4:0]	RW	0 0000	DMA base address This 5-bit vector defines the base-address for DMA transfers (when read/write access are done through the TIM3_DMAR address). DBA is defined as an offset starting from the address of the TIM1_CR1 register. Example: 00000: TIMx_CR1, 00001: TIMx_CR2, 00010: TIMx_SMCR, ...

18.4.18. TIM3 DMA address for full transfer (TIM3_DMAR)

Address offset: 0x4C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMAB[31:16]															
RW															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DMAB[15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
31:0	DMAB[31:0]	RW	0	DMA register for burst accesses A read or write operation to the DMAR register accesses the register located at the address (TIMx_CR1 address) + (DBA + DMA index) x 4 where TIMx_CR1 address is the address of the control register 1, DBA is the DMA base address configured in TIMx_DCR register, DMA index is automatically controlled by the DMA transfer, and ranges from 0 to DBL (DBL configured in TIMx_DCR).

18.4.19. TIM3 register map

Offs	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

et																															
0x00	TIM3_CR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CKD[1:0]		ARPE	CMS[1:0]		DIR	OPM	URS	UDIS	CEN
	Reset value																						0	0	0	0	0	0	0	0	0	0
0x04	TIM3_CR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TI1S	MMS[2:0]			CCDS	Res.	Res.	Res.
	Reset value																								0	0	0	0	0			
0x08	TIM3_SMCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MSM	TS[2:0]			OCCS	SMS[2:0]		
	Reset value																								0	0	0	0	0	0	0	0
0x0C	TIM3_DIER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TDE	Res.	CC4DE	CC3DE	CC2DE	CC1DE	UDE	Res.	TIE	Res.	CC4IE	CC3IE	CC2IE	CC1IE	UIE
	Reset value																0		0	0	0	0	0		0		0	0	0	0	0
0x10	TIM3_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC4O	CC3O	CC2O	CC1O	Res.	Res.	TIF	Res.	CC4IF	CC3IF	CC2IF	CC1IF	UIF
	Reset value																		0	0	0	0	0			0		0	0	0	0	0
0x14	TIM3_EGR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TG	Res.	CC4G	CC3G	CC2G	CC1G	UG
	Reset value																									0		0	0	0	0	0
0x18	TIM3_CCMR1(output compare mode)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC2CE	OC2M[2:0]			OC2PE	CO2FE	CC2S[1:0]		OC1CE	OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x18	TIM3_CCMR1(Input Capture mode)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IC2F[3:0]				IC2PSC[1:0]		CC2S[1:0]		IC1F[3:0]				IC1PSC[1:0]		CC1S[1:0]	
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x1C	TIM3_CCMR2(output compare mode)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC4CE	OC4M[2:0]			OC4PE	CO4FE	CC4S[1:0]		OC3CE	OC3M[2:0]			OC3PE	OC3FE	CC3S[1:0]	
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x1C	TIM3_CCMR2(Input	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IC4F[3:0]				IC4PSC[1:0]		CC4S[1:0]		IC3F[3:0]				IC3PSC[1:0]		CC3S[1:0]	

	Capture mode)																																
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x20	TIM3_CCER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC4NP	Res.	CC4P	CC4E	CC3NP	Res.	CC3P	CC3E	CC2NP	Res.	CC2P	CC2E	CC1NP	Res.	CC1P	CC1E
	Reset value																									0		0	0	0		0	0
0x24	TIM3_CNT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CNT[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x28	TIM3_PSC	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PSC[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x2C	TIM3_ARR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ARR[15:0]															
	Reset value																	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x34	TIM3_CCR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR1[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x38	TIM3_CCR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR2[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x3C	TIM3_CCR3	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR3[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x40	TIM3_CCR4	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR4[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x48	TIM3_DCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBL[4:0]					Res.	DBA[4:0]							
	Reset value																			0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x4C	TIM3_DMAR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DMAB[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

19. General-purpose timer (TIM14)

19.1. TIM14 introduction

The TIM14 general-purpose timer consists of a 16-bit auto-reload counter driven by a programmable prescaler. It may be used for a variety of purposes, including measuring the pulse lengths of input signals (input capture) or generating output waveforms (output compare, PWM).

Pulse lengths and waveform periods can be modulated from a few microseconds to several milliseconds using the timer prescaler and the RCC clock controller prescalers.

The TIM14 timer is completely independent, and does not share any resources. It can be synchronized together as described in TIM3.

19.2. TIM14 main features

- 16-bit auto-reload upcounter
- 16-bit programmable prescaler used to divide the counter clock frequency by any factor between 1 and 65535 (can be changed “on the fly”)
- One independent channel for:
  - Input capture
  - Output compare
  - PWM generation (edge-aligned mode)
- Interrupt generation on the following events:
  - Update: counter overflow, counter initialization (by software)
  - Input capture
  - Output compare

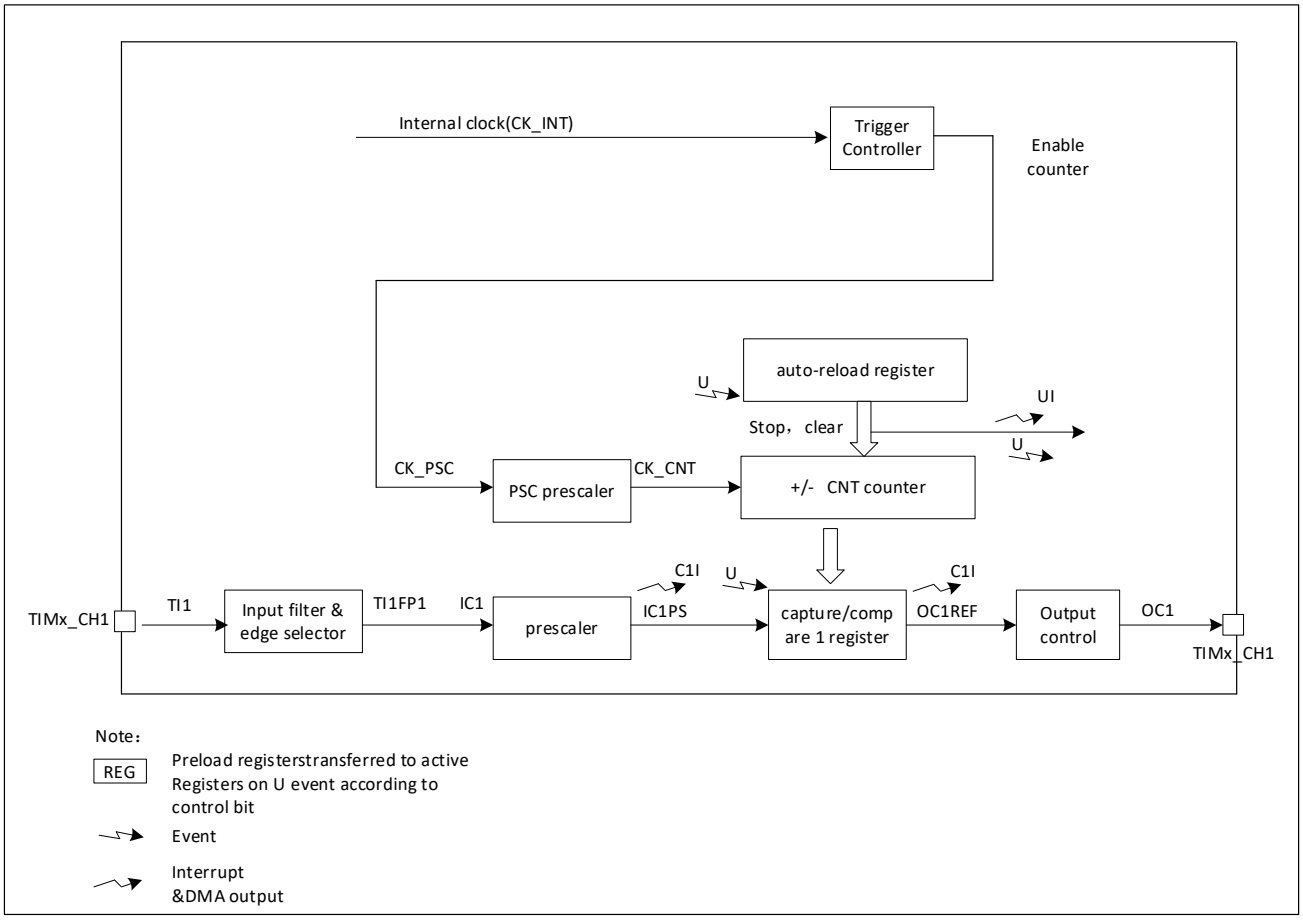

Figure 19-1 General-purpose timer block diagram (TIM14)

19.3. TIM14 functional description

19.3.1. Time-base unit

The main block of the programmable general-purpose timer is a 16-bit upcounter with its related auto-reload register. The counter clock can be divided by a prescaler.

The counter, the auto-reload register and the prescaler register can be written or read by software. This is true even when the counter is running.

The time-base unit includes:

- Counter register (TIM14_CNT)
- Prescaler register (TIM14_PSC)
- Auto-reload register (TIM14_ARR)

The auto-reload register is preloaded. Writing to or reading from the auto-reload register accesses the preload register. The content of the preload register are transferred into the shadow register permanently or at each update event (UEV), depending on the auto-reload preload enable bit (ARPE) in TIM14_CR1 register. The update event is sent when the counter reaches the overflow and if the UDIS bit equals 0 in the TIM14_CR1 register. It can also be generated by software.

The counter is clocked by the prescaler output CK_CNT, which is enabled only when the counter enable bit (CEN) in TIM14_CR1 register is set.

Note that the counter starts counting 1 clock cycle after setting the CEN bit in the TIM14_CR1 register.

Prescaler description

The prescaler can divide the counter clock frequency by any factor between 1 and 65535. It is based on a 16-bit counter controlled through a 16-bit register (in the TIM14_PSC register). It can be changed on the fly as this control register is buffered. The new prescaler ratio is taken into account at the next update event.

The following figures give some examples of the counter behavior when the prescaler ratio is changed on the fly.

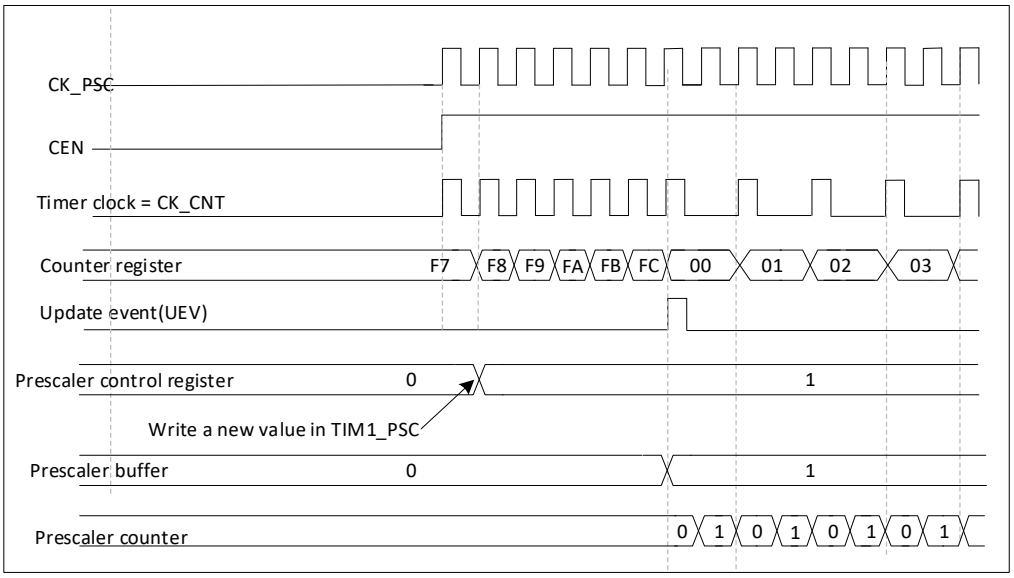

Figure 19-2 Counter timing diagram with prescaler division change from 1 to 2

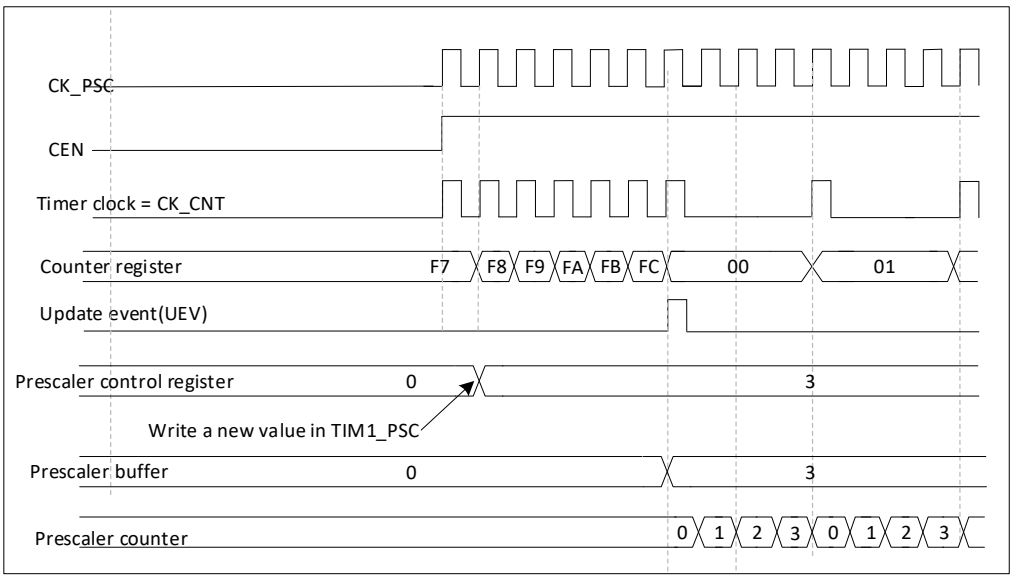

Figure 19-3 Counter timing diagram with prescaler division change from 1 to 4

Upcounter mode

The counter counts from 0 to the auto-reload value (content of the TIM14_ARR register), then restarts from 0 and generates a counter overflow event.

Every time the count overflows, an update event is generated. Setting the UG bit in the TIM14_EGR register also generates an update event.

The UEV event can be disabled by software by setting the UDIS bit in the TIM14_CR1 register. This is to avoid updating the shadow registers while writing new values in the preload registers. Then no update event occurs until the UDIS bit has been written to 0. However, the counter restarts from 0, as well as the counter of the prescaler (but the prescale rate does not change). In addition, if the URS bit (update request selection) in TIM14_CR1 register is set, setting the UG bit generates an update event UEV but without setting the UIF flag (thus no interrupt or DMA request is sent). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

When an update event occurs, all the registers are updated and the update flag (UIF bit in TIM14_SR register) is set (depending on the URS bit):

- The auto-reload shadow register is updated with the preload value (TIM14_ARR),
- The buffer of the prescaler is reloaded with the preload value (content of the TIM14_PSC register).

The following figures show some examples of the counter behavior for different clock frequencies when TIMx_ARR = 0x36.

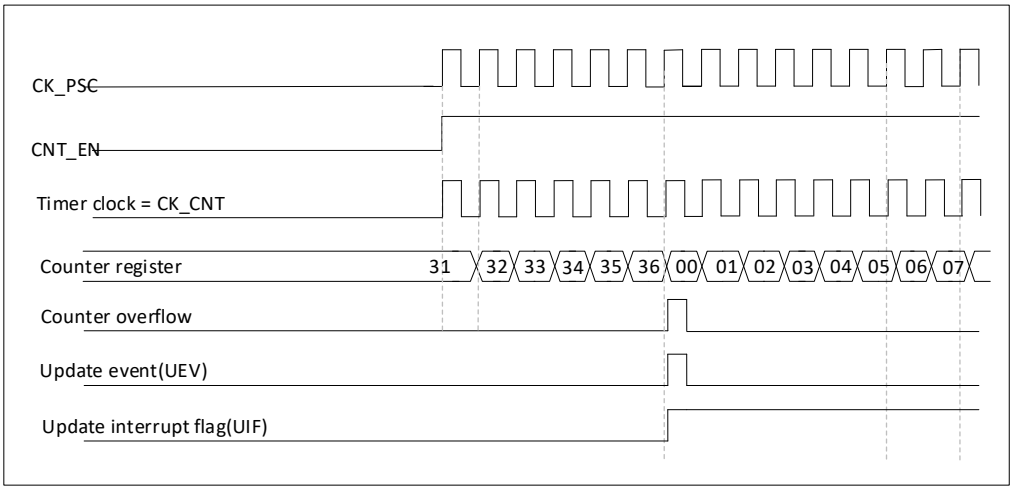

Figure 19-4 Counter timing diagram, internal clock divided by 1

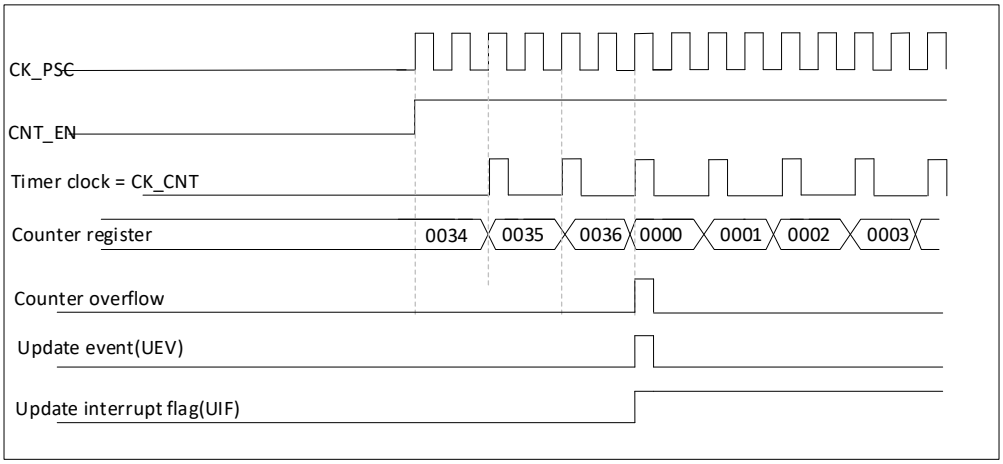

Figure 19-5 Counter timing diagram, internal clock divided by 2

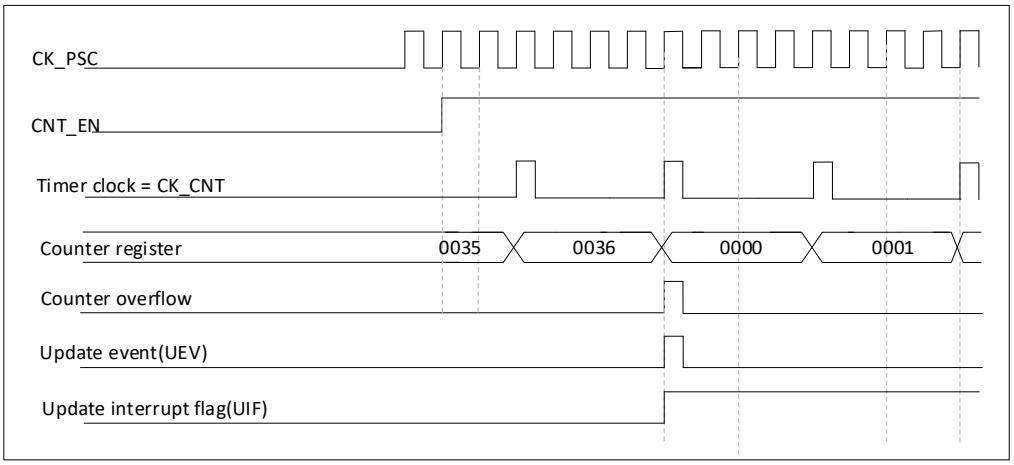

Figure 19-6 Counter timing diagram, internal clock divided by 4

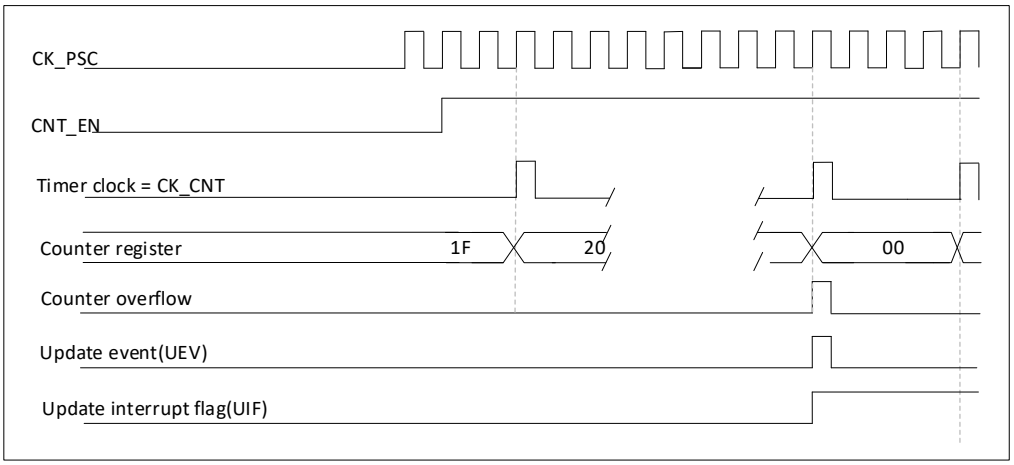

Figure 19-7 Counter timing diagram, internal clock divided by N

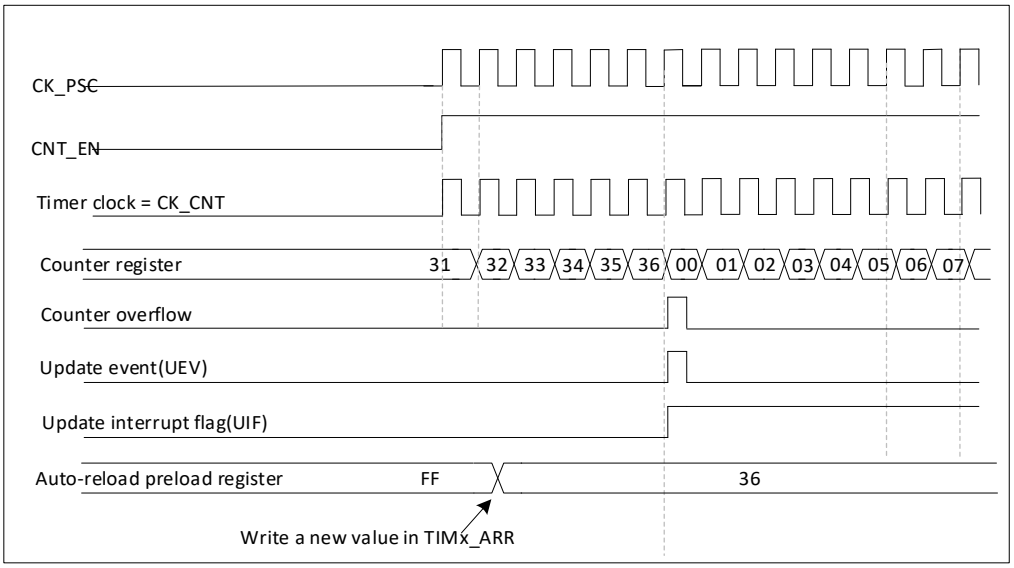

Figure 19-8 Counter timing diagram, update event when ARPE = 0 (TIMx_ARR not preloaded)

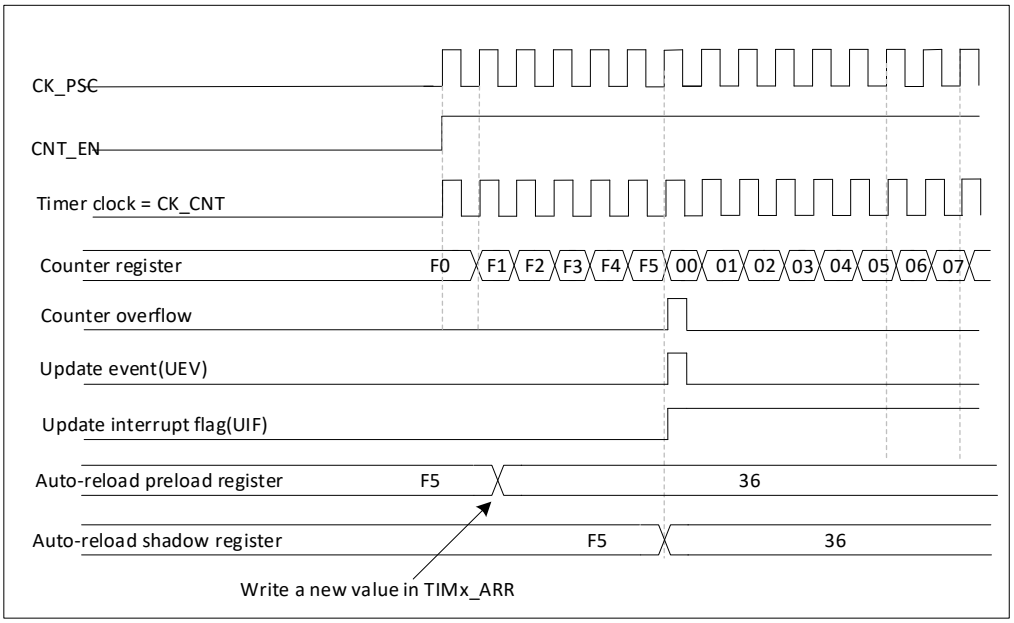

Figure 19-9 Counter timing diagram, update event when ARPE = 1 (TIMx_ARR preloaded)

19.3.2. Clock source

The counter clock is provided by the Internal clock (CK_INT) source. The CEN (in the TIMx_CR1 register) and UG bits (in the TIM14_EGR register) are actual control bits and can be changed only by software (except for UG that remains cleared automatically). As soon as the CEN bit is written to 1, the prescaler is clocked by the internal clock CK_INT.

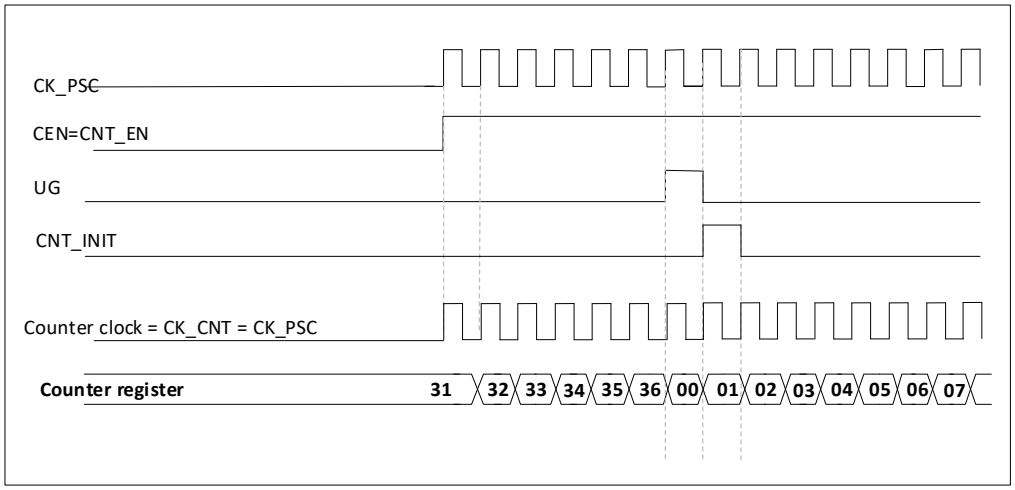

Figure 19-10 Control circuit in normal mode, internal clock divided by 1

19.3.3. Capture/compare channels

Each Capture/Compare channel is built around a capture/compare register (including a shadow register), a input stage for capture (with digital filter, multiplexing and prescaler) and an output stage (with comparator and output control).

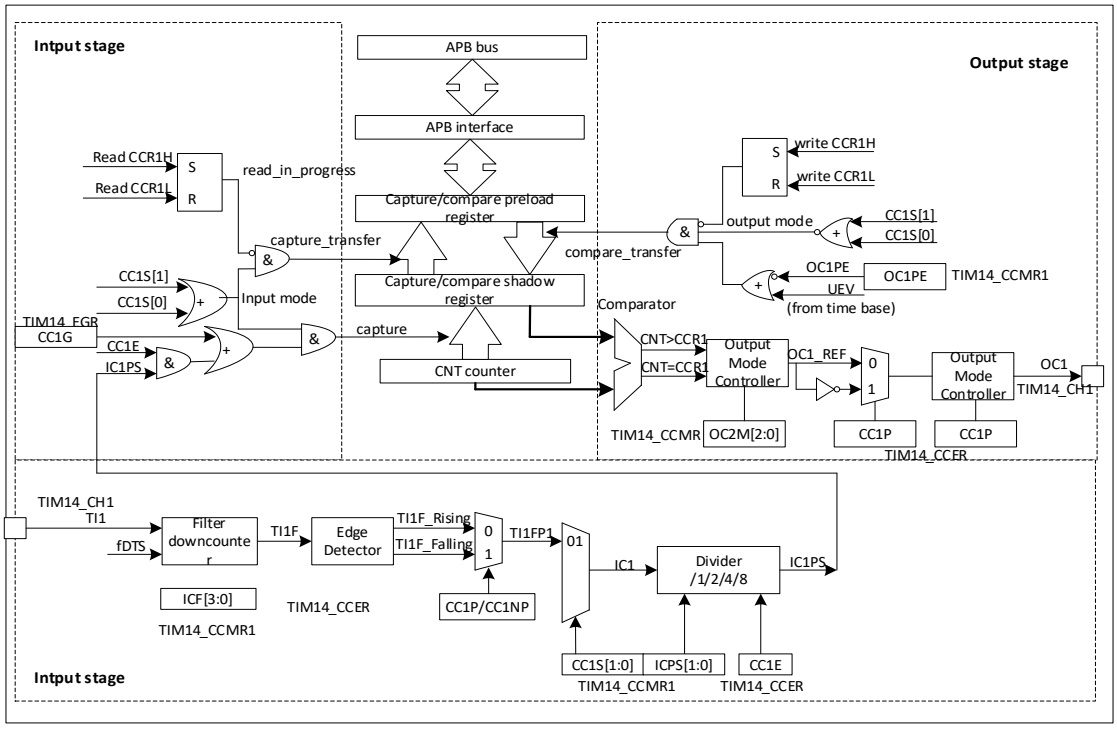

Figure 19-11 TIM14 Capture/compare channel

The input stage samples the corresponding TIx input to generate a filtered signal TIxF. Then, an edge detector with polarity selection generates a signal (TIxFPx) which can be used as trigger input by the slave mode controller or as the capture command. It is prescaled before the capture register (ICxPS).

The output stage generates an intermediate waveform which is then used for reference: OCxRef (active high). The polarity acts at the end of the chain.

The capture/compare block is made of one preload register and one shadow register. Write and read only access the preload register.

In capture mode, captures are actually done in the shadow register, which is copied into the preload register.

In compare mode, the content of the preload register is copied into the shadow register which is compared to the counter

19.3.4. Input capture mode

In Input capture mode, the Capture/Compare Registers (TIM14_CCRx) are used to latch the value of the counter after a transition detected by the corresponding ICx signal. When a capture occurs, the corresponding CCXIF flag (TIM14_SR register) is set. If a capture occurs while the CCxIF flag was already high, then the over-capture flag CCxOF (TIMx_SR register) is set. CCxIF can be cleared by software by writing it to ‘0’ or by reading the captured data stored in the TIMx_CCRx register. CCxOF is cleared when it is written with 0.

The following example shows how to capture the counter value in TIM14_CCR1 when TI1 input rises. To do this, use the following procedure:

- Select the active input: TIM14_CCR1 must be linked to the TI1 input, so write the CC1S bits to ‘01’ in the TIM14_CCMR1 register. As soon as CC1S becomes different from ‘00’, the channel is configured in input mode and the TIM14_CCR1 register becomes readonly.
- Program the appropriate input filter duration in relation with the signal connected to the timer (when the input is one of the TIx (ICxF bits in the TIM14_CCMRx register). When toggling, the input signal is not stable

during at must 5 internal clock cycles. We must program a filter duration longer than these 5 clock cycles. We can validate a transition on TI1 when 8 consecutive samples with the new level have been detected (sampled at fDTS frequency). Then write IC1F bits to '0011' in the TIM14_CCMR1 register.

- Select the edge of the active transition on the TI1 channel by programming CC1P and CC1NP bits to '00' in the TIMx_CCER register (rising edge in this case).
- Program the input prescaler. In our example, we wish the capture to be performed at each valid transition, so the prescaler is disabled (write IC1PS bits to '00' in the TIMx_CCMR1 register).
- Enable capture from the counter into the capture register by setting the CC1E bit in the TIMx_CCER register.
- If needed, enable the related interrupt request by setting the CC1IE bit in the TIMx_DIER register.

When an input capture occurs:

- The TIM14_CCR1 register gets the value of the counter on the active transition
- CC1IF flag is set (interrupt flag). CC1OF is also set if at least two consecutive captures occurred whereas the flag was not cleared.
- An interrupt is generated depending on the CC1IE bit.
- In order to handle the overcapture, it is recommended to read the data before the overcapture flag. This is to avoid missing an overcapture which could happen after reading the flag and before reading the data.

Note: IC interrupt requests can be generated by software by setting the corresponding CCxG bit in the TIMx_EGR register.

19.3.5. Forced output mode

In output mode (CCxS bits = '00' in the TIM14_CCMRx register), each output compare signal (OCxREF and then OCx) can be forced to active or inactive level directly by software, independently of any comparison between the output compare register and the counter.

To force an output compare signal (OCXREF/OCx) to its active level, one just needs to write '101' in the OCxM bits in the corresponding TIM14_CCMRx register. Thus OCXREF is forced high (OCxREF is always active high) and OCx get opposite value to CCxP polarity bit.

For example: CCxP = '0' (OCx active high) = > OCx is forced to high level.

The OCxREF signal can be forced low by writing the OCxM bits to '100' in the TIM14_CCMRx register.

The comparison between the TIM14_CCRx shadow register and the counter is still performed and allows the flag to be set. Interrupt requests can be sent accordingly. This is described in the output compare mode section below.

19.3.6. Output compare mode

This function is used to control an output waveform or to indicate when a period of time has elapsed.

When a match is found between the capture/compare register and the counter, the output compare function:

- Assigns the corresponding output pin to a programmable value defined by the output compare mode (OCxM bits in the TIM14_CCMRx register) and the output polarity (CCxP bit in the TIMx_CCER register). The output pin can keep its level (OCXM = '000'), be set active (OCxM = '001'), be set inactive (OCxM = '010') or can toggle (OCxM = '011') on match.
- Sets a flag in the interrupt status register (CCxIF bit in the TIMx_SR register).
- Generates an interrupt if the corresponding interrupt mask is set (CCxIE bit in the TIM14_DIER register).

The TIM14_CCRx registers can be programmed with or without preload registers using the OCxPE bit in the TIM14_CCMRx register.

In output compare mode, the update event UEV has no effect on OCxREF and OCx output. The timing resolution is one count of the counter. Output compare mode can also be used to output a single pulse (in One-pulse mode).(This is meaningless, no OPM)

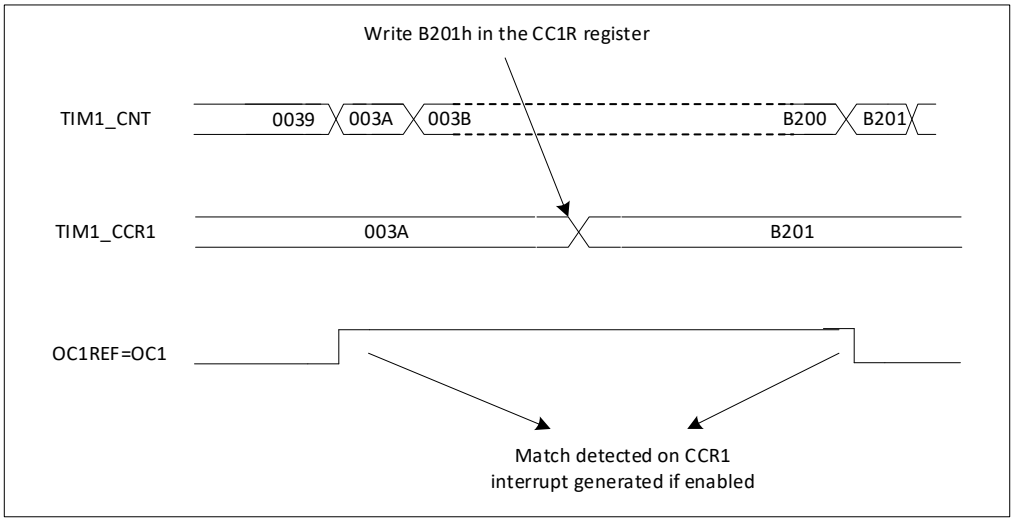

Figure 19-12 Output compare mode, toggle on OC1

19.3.7. PWM mode

Pulse Width Modulation mode allows to generate a signal with a frequency determined by the value of the TIMx_ARR register and a duty cycle determined by the value of the TIMx_CCRx register.

The PWM mode can be selected independently on each channel (one PWM per OCx output) by writing '110' (PWM mode 1) or '111' (PWM mode 2) in the OCxM bits in the TIM14_CCMRx register. The corresponding preload register must be enabled by setting the OCxPE bit in the TIM14_CCMRx register, and eventually the auto-reload preload register by setting the ARPE bit in the TIM14_CR1 register.

As the preload registers are transferred to the shadow registers only when an update event occurs, before starting the counter, all registers must be initialized by setting the UG bit in the TIM14_EGR register.

The OCx polarity is software programmable using the CCxP bit in the TIM14_CCER register. It can be programmed as active high or active low. The OCx output is enabled by the CCxE bit in the TIM14_CCER register.

In PWM mode (1 or 2), TIM14_CNT and TIM14_CCRx are always compared to determine whether TIM14_CNT≤TIM14_CCRx.

The timer is able to generate PWM in edge-aligned mode only since the counter is upcounting.

PWM edge-aligned mode

In the following example, we consider PWM mode 1. The reference PWM signal OCxREF is high as long as TIM14_CNT < TIM14_CCRx else it becomes low. If the compare value in TIM14_CCRx is greater than the auto-reload value (in TIM14_ARR) then OCxREF is held at'1'. If the compare value is 0 then OCxRef is held at '0'.

The following figure shows some edgealigned PWM waveforms in an example where TIMx_ARR = 8.

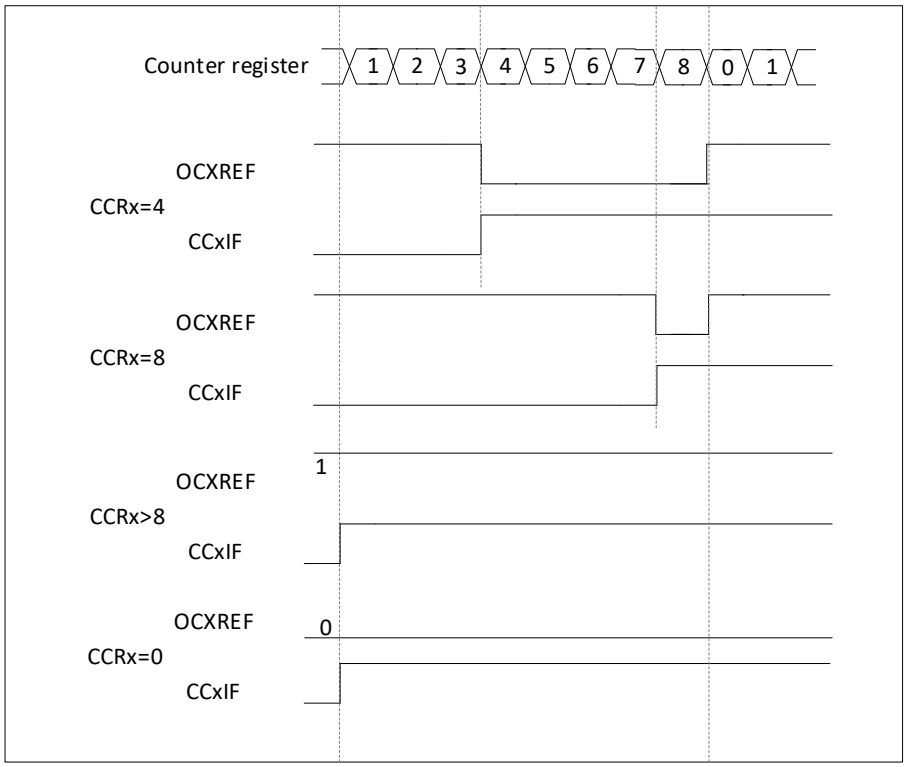

Figure 19-13 Edge-aligned PWM waveforms (ARR = 8)

19.3.8. Debug mode

When the microcontroller enters debug mode (M0+ core halted), the TIMx counter either continues to work normally or stops, depending on DBG_TIMx_STOP configuration bit in DBG module.

19.4. TIM14 registers

19.4.1. TIM14 control register 1 (TIM14_CR1)

Address offset: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	CKD[1:0]		ARPE	Res		Res	Res	URS	UDIS	CEN
-	-	-	-	-	-	RW		RW	-		-	-	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	0	Reserved, must be kept at reset value.
9:8	CKD[1:0]	RW	00	Clock division This bit-field indicates the division ratio between the timer clock (CK_INT) frequency and sampling clock used by the digital filters (ETR, TIx), 00: tDTS = tCK_INT 01: tDTS = 2 × tCK_INT 10: tDTS = 4 × tCK_INT 11: Reserved
7	ARPE	RW	0	Auto-reload preload enable 0: TIMx_ARR register is not buffered 1: TIMx_ARR register is buffered
6:3	Reserved	-	0	Reserved, must be kept at reset value.
2	URS	RW	0	Update request source

				This bit is set by software to select the update interrupt (UEV) sources. 0: Any of the following events generate an UEV or a DMA request if enabled: – Counter overflow – Setting the UG bit 1: Only counter overflow generates an UEV or a DMA request if enabled.
1	UDIS	RW	0	Update disable This bit is set and cleared by software to enable/disable update interrupt (UEV) event generation. 0: UEV enabled. An UEV is generated by one of the following events: – Counter overflow – Setting the UG bit. Buffered registers are then loaded with their preload values. 1: UEV disabled. No UEV is generated, shadow registers keep their value (ARR, PSC, CCRx). The counter and the prescaler are reinitialized if the UG bit is set.
0	CEN	RW	0	Counter enable 0: Counter disabled 1: Counter enabled Note: External clock, gated mode and encoder mode can only work after the CEN bit is set by software. Trigger mode can automatically set the CEN bit by hardware.

19.4.2. TIM14 DMA/interrupt enable register (TIM14_DIER)

Address offset: 0x0C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res														CC1IE	UIE
-														RW	RW

Bit	Name	R/W	Reset Value	Function
31:2	Reserved			Reserved, must be kept at reset value.
1	CC1IE	RW	0	CC1IE: Capture/Compare 1 interrupt enable 0: CC1 interrupt disabled 1: CC1 interrupt enabled
0	UIE	RW	0	UIE: Update interrupt enable 0: Update interrupt disabled 1: Update interrupt enabled

19.4.3. TIM14 status register (TIM14_SR)

Address offset: 0x010

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res						CC1OF	Res							CC1F	UIF
-						Rc_w0	-							Rc_w0	Rc_w0

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	0	Reserved, must be kept at reset value.
9	CC1OF	Rc_w0	0	Capture/Compare 1 overcapture flag This flag is set by hardware only when the corresponding channel is configured in input capture mode. It is cleared by software by writing it to ‘0’.

				0: No overcapture has been detected. 1: The counter value has been captured in TIM14_CCR1 register while CC1IF flag was already set
8:2	Res	Rc_w0	0	Reserved, must be kept at reset value.
1	CC1IF	Rc_w0	0	Capture/compare 1 interrupt flag Condtion: channel CC1 is configured as output This flag is set by hardware when the counter matches the compare value. It is cleared by software. 0: No match. 1: The content of the counter TIM14_CNT matches the content of the TIM14_CCR1 register. Condition: channel CC1 is configured as input This bit is set by hardware on a capture. It is cleared by software or by reading the TIM14_CCR1 register. 0: No input capture occurred. 1: The counter value has been captured in TIM14_CCR1 register (an edge has been detected on IC1 which matches the selected polarity).
0	UIF	Rc_w0	0	Update interrupt flag This bit is set by hardware on an update event. It is cleared by software. 0: No update occurred. 1: Update interrupt pending. This bit is set by hardware when the registers are updated: – At overflow and if UDIS = ’0’ in the TIMx_CR1 register. – When CNT is reinitialized by software using the UG bit in TIMx_EGR register, if URS = ’0’ and UDIS = ’0’ in the TIMx_CR1 register.

19.4.4. TIM14 event generation register (TIM14_EGR)

Address offset: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res														CC1G	UG
-														W	W

Bit	Name	R/W	Reset Value	Function
31:2	Reserved	-	0	Reserved, must be kept at reset value.
1	CC1G	W	0	Capture/compare 1 generation This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: A capture/compare event is generated on channel 1: Condition: channel CC1 is configured as output CC1IF flag is set, Corresponding interrupt or DMA request is sent if enabled. Condition: channel CC1 is configured as input The current value of the counter is captured in TIMx_CCR1 register. The CC1IF flag is set, the corre-sponding interrupt is sent if enabled. The CC1OF flag is set if the CC1IF flag was already high.
0	UG	W	0	Update generation This bit can be set by software, it is automatically cleared by hardware. 0: No action 1: Re-initialize the counter and generates an update of the registers. Note that the prescaler counter is cleared too (anyway the prescaler ratio is not affected). The counter is cleared.

19.4.5. TIM14 capture/compare mode register 1 (TIM14_CCMR1)

Address offset: 0x18

Reset value: 0x0000 0000

output compare mode:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res								Res	OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	
-								-	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:7	Reserved	-		Reserved, must be kept at reset value.
6:4	OC1M[2:0]	RW	00	Output compare 1 mode These bits define the behavior of the output reference signal OC1REF from which OC1, OC1N is derived. OC1REF is active high whereas OC1, OC1N active level depends on CC1P, CC1NP bit. 000: Frozen. The comparison between the output compare register TIMx_CCR1 and the counter TIMx_CNT has no effect on the outputs. 001: Set channel 1 to active level on match. OC1REF signal is forced high when the counter TIMx_CNT matches the capture/compare register 1 (TIMx_CCR1). 010: Set channel 1 to inactive level on match. OC1REF signal is forced low when the counter TIMx_CNT matches the capture/compare register 1 (TIMx_CCR1). 011: Toggle - OC1REF toggles when TIMx_CNT = TIMx_CCR1. 100: Force inactive level - OC1REF is forced low. 101: Force active level - OC1REF is forced high. 110: PWM mode 1 - when upcounting, Channel 1 is active as long as TIMx_CNT < TIMx_CCR1 else inactive. when downcounting, Channel 1 is inactive(OC1REF = 0) as long as TIMx_CNT > TIMx_CCR1 else active(OC1REF = 1). 111: PWM mode 2 - Channel 1 is inactive as long as TIMx_CNT < TIMx_CCR1 else active. Note: In PWM mode 1 or 2, the OCREF level changes when the result of the comparison changes or when the output compare mode switches from frozen to PWM mode.
3	OC1PE	RW	0	Output compare 1 preload enable 0: Preload register on TIM14_CCR1 disabled. TIM14_CCR1 can be written at anytime, the new value is taken in account immediately. 1: Preload register on TIM14_CCR1 enabled. Read/Write operations access the preload register. TIM14_CCR1 preload value is loaded in the active register at each update event.
2	OC1FE	RW	0	Output compare 1 fast enable This bit is used to accelerate the effect of an event on the trigger in input on the CC output. 0: CC1 behaves normally depending on counter and CCR1 values even when the trigger is ON. The minimum delay to activate CC1 output when an edge occurs on the trigger input is 5 clock cycles. 1: An active edge on the trigger input acts like a compare match on CC1 output. OC is then set to the compare level independently of the result of the comparison. Delay to sample the trigger input and to activate CC1 output is reduced to 3 clock cycles. OC1FE acts only if the channel is configured in PWM1 or PWM2 mode.
1:0	CC1S[1:0]	RW	00	Capture/Compare 1 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC1 channel is configured as output. 01: CC1 channel is configured as input, IC1 is mapped on TI1. 10: Reserved

				11: Reserved Note: CC1S bits are writable only when the channel is OFF (CC1E = 0 in TIM14_CCER).

Input Capture mode:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res								IC1F[3:0]				IC1PSC[1:0]		CC1S[1:0]	
-								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-		Reserved, must be kept at reset value.
7:4	IC1F[3:0]	RW	0000	Input capture 1 filter This bit-field defines the frequency used to sample TI1 input and the length of the digital filter applied to TI1. The digital filter is made of an event counter in which N events are needed to validate a transition on the output: 0000: No filter, sampling is done at fDTS 0001: fSAMPLING = fCK_INT, N = 2 0010: fSAMPLING = fCK_INT, N = 4 0011: fSAMPLING = fCK_INT, N = 8 0100: fSAMPLING = fDTS / 2, N = 6 0101: fSAMPLING = fDTS / 2, N = 8 0110: fSAMPLING = fDTS / 4, N = 6 0111: fSAMPLING = fDTS / 4, N = 8 1000: fSAMPLING = fDTS / 8, N = 6 1001: fSAMPLING = fDTS / 8, N = 8 1010: fSAMPLING = fDTS / 16, N = 5 1011: fSAMPLING = fDTS / 16, N = 6 1100: fSAMPLING = fDTS / 16, N = 8 1101: fSAMPLING = fDTS / 32, N = 5 1110: fSAMPLING = fDTS / 32, N = 6 1111: fSAMPLING = fDTS / 32, N = 8
3:2	IC1PSC[1:0]	RW	00	Input capture 1 prescaler This bit-field defines the ratio of the prescaler acting on CC1 input (IC1). The prescaler is reset as soon as CC1E = ’0’ (TIM1_CCER register). 00: no prescaler, capture is done each time an edge is detected on the capture input 01: capture is done once every 2 events 10: capture is done once every 4 events 11: capture is done once every 8 events
1:0	CC1S[1:0]	RW	00	Capture/Compare 1 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC1 channel is configured as output 01: CC1 channel is configured as input, IC1 is mapped on TI1 Other: Reserved Note: CC1S bits are writable only when the channel is OFF (CC1E = 0 in TIM14_CCER).

19.4.6. TIM14 capture/compare enable register (TIM14_CCER)

Address offset: 0x20

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	CC1NP	Res	CC1P	CC1E
												RW		RW	RW

Bit	Name	R/W	Reset Value	Function
31:4	Reserved	-	0	Reserved, must be kept at reset value.
3	CC1NP	RW	0	Input/Capture 1 complementary output Polarity. CC1 channel configured as output: CC1NP must be kept cleared. CC1 channel configured as input: CC1NP bit is used in conjunction with CC1P to define TI1FP1 polarity (refer to CC1P description).
2	Reserved	-	0	Reserved, must be kept at reset value.
1	CC1P	RW	0	Input/Capture 1 output Polarity. Condition: CC1 channel configured as output 0: OC1 active high 1: OC1 active low Condition: CC1 channel configured as input The CC1NP/CC1P bits select TI1FP1 and TI2FP1 polarity for trigger or capture operations. 00: noninverted/rising edge Circuit is sensitive to TI1FP1 rising edge (capture mode, reset trigger, external clock or trigger mode), TI1FP1 is not inverted(gate mode, code mode). 01: inverted/falling edge Circuit is sensitive to TI1FP1 falling edge (capture mode, reset trigger, external clock or trigger mode), TI1FP1 is inverted(gate mode, code mode). 10: reserved, do not use this configuration. 11: noninverted/both edges
0	CC1E	RW	0	Input/Capture 1 output enable. Condition: CC1 channel configured as output: 0: Off - OC1 is not active 1: On - OC1 signal is output on the corresponding output pin Condition: CC1 channel configured as input: This bit determines if a capture of the counter value can actually be done into the input capture register 1 (TIMx_CCR1) or not. 0: Capture disabled 1: Capture enabled

CcxE bit	OCx output State
0	Output Disabled (OCx = 0,OCx_EN = 0)
1	OCx = OCxREF+Polarity,OCx_EN = 1

19.4.7. TIM14 counter (TIM14_CNT)

Address offset: 0x24

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CNT[15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			Reserved, must be kept at reset value.
15:0	CNT[15:0]	RW	0	Counter value

19.4.8. TIM14 prescaler (TIM14_PSC)

Address offset: 0x28

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**

PSC[15:0]
RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			Reserved, must be kept at reset value.
15:0	PSC[15:0]	RW	0	Prescaler value The counter clock frequency CK_CNT is equal to fCK_PSC / (PSC[15:0] + 1). PSC contains the value to be loaded in the active prescaler register at each update event. Cleared to 0 by the UG bit in TIM_EGR or by a slave controller operating in reset mode.

19.4.9. TIM14 auto-reload register (TIM14_ARR)

Address offset: 0x2C

Reset value: 0x0000 FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ARR[15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			Reserved, must be kept at reset value.
15:0	ARR[15:0]	RW	0	Auto-reload value ARR is the value to be loaded in the actual auto-reload register. Refer to the Time-base unit for more details about ARR update and behavior. The counter is blocked while the auto-reload value is null.

19.4.10. TIM14 capture/compare register 1 (TIM14_CCR1)

Address offset: 0x34

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR1[15:0]															
RW/RO															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			Reserved, must be kept at reset value.
15:0	CCR1[15:0]	RW	0	Capture/Compare 1 value Condition: channel CC1 is configured as output CCR1 is the value to be loaded in the actual capture/compare 1 register (preload value). It is loaded permanently if the preload feature is not selected in the TIM1_CCMR1 register (bit OC1PE). Else the preload value is copied in the active capture/compare 1 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC1 output. Condition: channel CC1is configured as input CCR1 is the counter value transferred by the last input capture 1 event (IC1).

19.4.11. TIM14 option register (TIM14_OR)

Address offset: 0x50

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
-															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res														TI1_RMP	
-														RW	RW

Bit	Name	R/W	Reset Value	Function
31:2	Reserved	-	0	Reserved, must be kept at reset value.
1:0	TI1_RMP	RW	0	Timer Input 1 remap Set and cleared by software. 00: TIM14 Channel1 is connected to the GPIO. Refer to the alternate function mapping in the device datasheets. 01: TIM14 Channel1 is connected to the RTCCLK. 10: TIM14 Channel1 is connected to the HSE/32 Clock. 11: TIM14 Channel1 is connected to the microcontroller clock output (MCO), this selection is controlled by the MCO[2:0] bits of the Clock configuration register (RCC_CFGR)

19.4.12. TIM14 register map

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x000	TIM14_CR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CKD[1:0]		ARPE	Res.		Res.	Res.	URS	UDIS	CEN
	Reset value																							0	0	0					0	0	0
0x00C	TIM14_DIER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC1IE	UIE
	Reset value																															0	0
0x010	TIM14_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC1O	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC1IF	UIF
	Reset value																							0								0	0
0x014	TIM14_EGR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC1G	UG
	Reset value																															0	0
0x018	TIM14_CCMR1(output compare mode)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC1M[2:0]				OC1PE	OC1FE	CC1S[1:0]	
	Reset value																										0	0	0		0	0	0	0
0x018	TIM14_CCMR1(Input Capture mode)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IC1F[3:0]				IC1PSC[1:0]		CC1S[1:0]	
	Reset value																									0	0	0	0	0	0	0	0

0x20	TIM14_CCER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC1NP	Res.	CC1P	CC1E
	Reset value																													0		0	0
0x24	TIM14_CNT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CNT[15:0]																				
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x28	TIM14_PSC	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PSC[15:0]																				
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x2C	TIM14_ARR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ARR[15:0]																				
	Reset value																	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x34	TIM14_CCR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR1[15:0]																				
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x50	TIMx_OR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TI1_RMP[1:0]			
	Reset value																															0	0		

20. General-purpose timers (TIM16/17)

TIM16 and TIM17 function exactly the same.

20.1. TIM16 and TIM17 main features

- 16-bit auto-reload upcounter
- 16-bit programmable prescaler used to divide (also “on the fly”) the counter clock frequency by any factor between 1 and 65536
- One channel for:
  - Input capture
  - Output compare
  - PWM generation (Edge-aligned mode)
  - One-pulse mode output
- Complementary outputs with programmable dead-time
- Repetition counter to update the timer registers only after a given number of cycles of the counter
- Break input to put the timer’s output signals in the reset state or a known state
- Interrupt/DMA generation on the following events:
  - Update: counter overflow
  - Input capture
  - Output compare
  - Break input

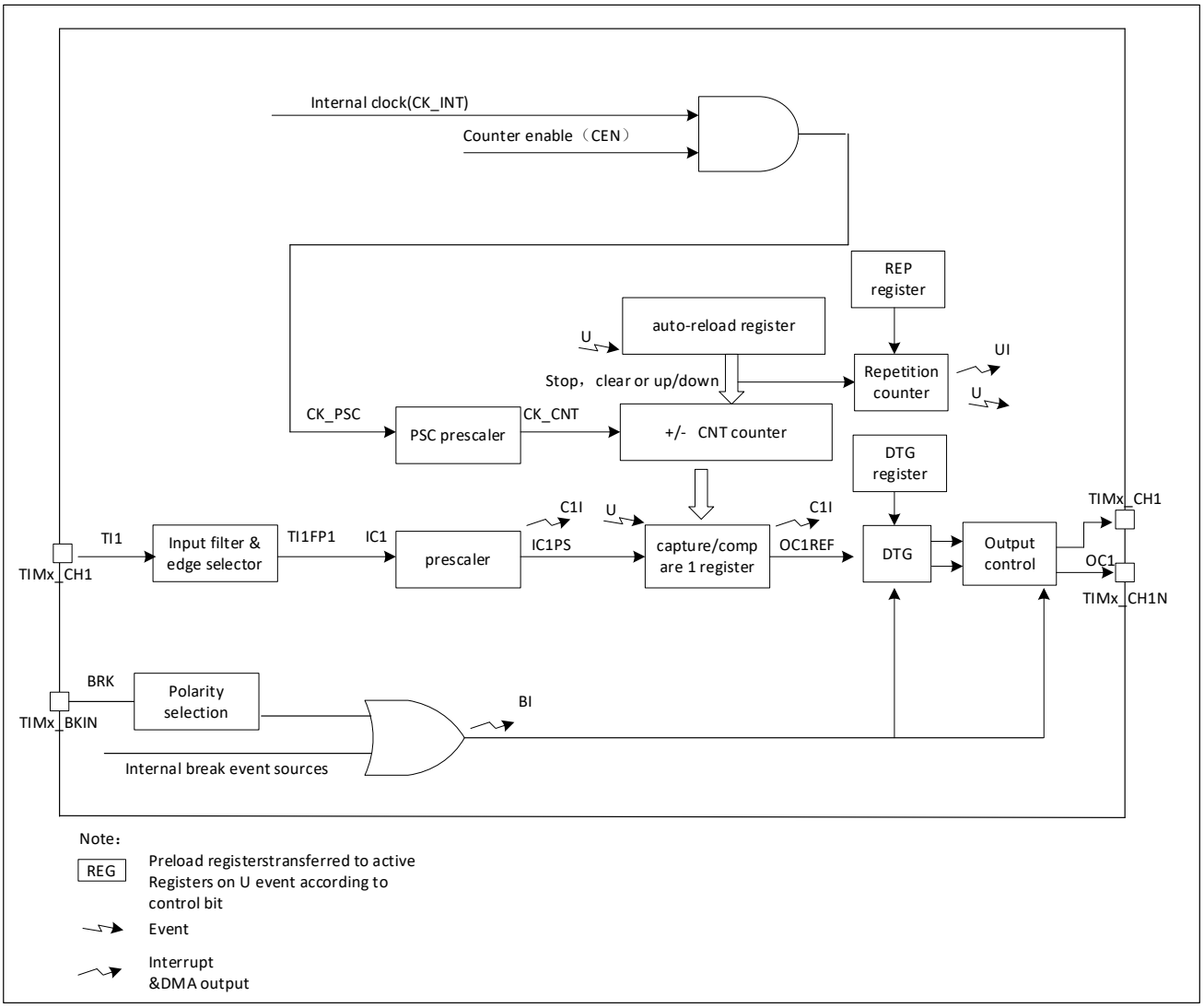

Figure 20-1 TIM16 and TIM17 block diagram

20.2. TIM16/TIM17 functional description

20.2.1. Time-base unit

The main block of the programmable general purpose timer is a 16-bit upcounter with its related auto-reload register. The counter clock can be divided by a prescaler.

The counter, the auto-reload register and the prescaler register can be written or read by software. This is true even when the counter is running.

The time-base unit includes:

- Counter register (TIMx_CNT)
- Prescaler register (TIMx_PSC)
- Auto-reload register (TIMx_ARR)
- Repetition counter register (TIMx_RCR)

The auto-reload register is preloaded. Writing to or reading from the auto-reload register accesses the preload register. The content of the preload register are transferred into the shadow register permanently or at each update event (UEV), depending on the auto-reload preload enable bit (ARPE) in TIMx_CR1 register. The update

event is sent when the counter reaches the overflow and if the UDIS bit equals 0 in the TIMx_CR1 register. It can also be generated by software.

The counter is clocked by the prescaler output CK_CNT, which is enabled only when the counter enable bit (CEN) in TIMx_CR1 register is set.

Note that the counter starts counting 1 clock cycle after setting the CEN bit in the TIMx_CR1 register.

Prescaler description

The prescaler can divide the counter clock frequency by any factor between 1 and 65535. It is based on a 16-bit counter controlled through a 16-bit register (in the TIMx_PSC register). It can be changed on the fly as this control register is buffered. The new prescaler ratio is taken into account at the next update event.

The following figures give some examples of the counter behavior when the prescaler ratio is changedy:

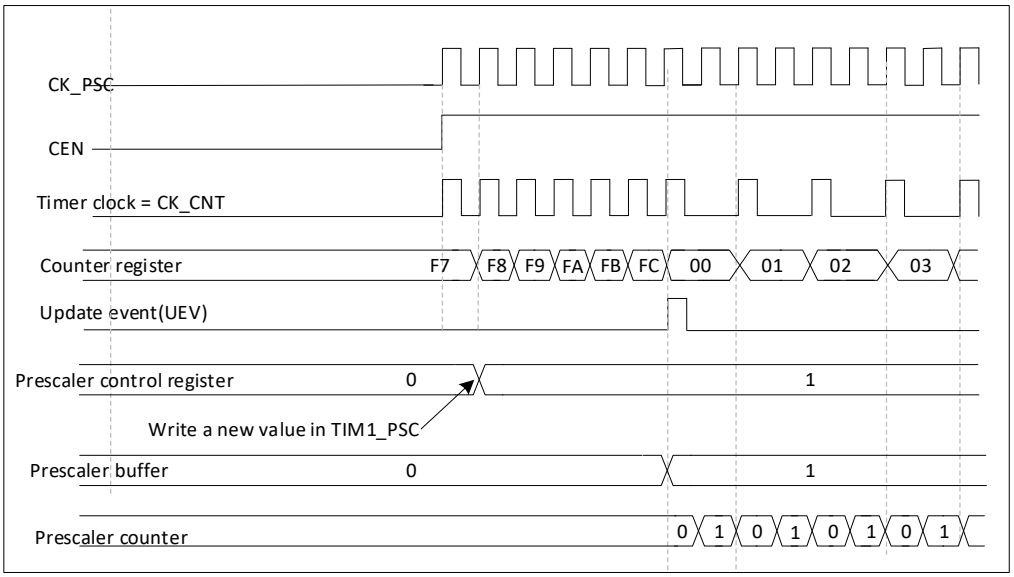

Figure 20-2 Counter timing diagram with prescaler division change from 1 to 2

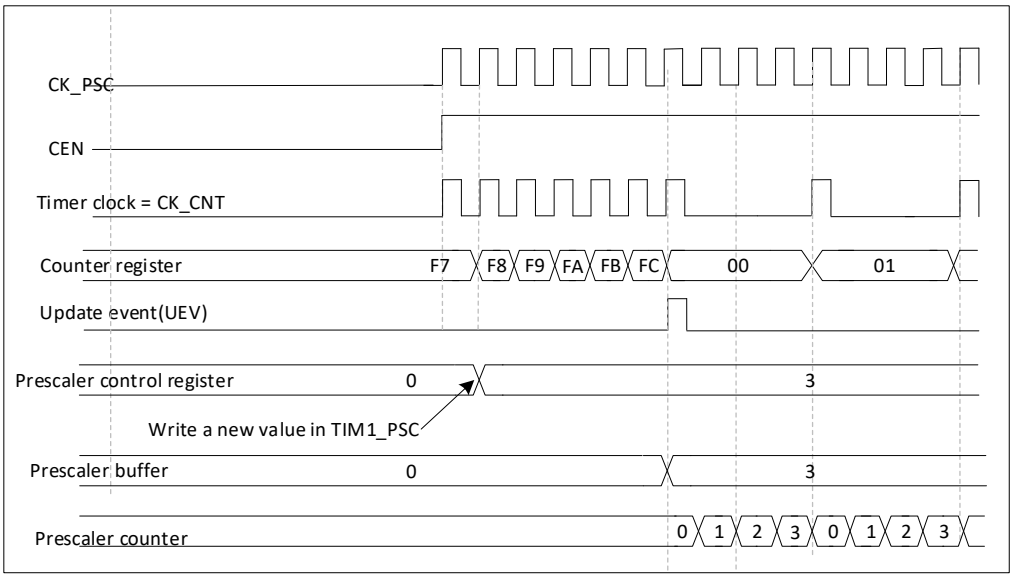

Figure 20-3 Counter timing diagram with prescaler division change from 1 to 4

20.2.2. Counter operation

The counter counts from 0 to the auto-reload value (content of the TIMx_ARR register), then restarts from 0 and generates a counter overflow event.

If the repetition counter is used, the update event (UEV) is generated after upcounting is repeated for the number of times programmed in the repetition counter register (TIMx_RCR). Else the update event is generated at each counter overflow.

Setting the UG bit in the TIMx_EGR register (by software or by using the slave mode controller) also generates an update event.

The UEV event can be disabled by software by setting the UDIS bit in the TIMx_CR1 register. This is to avoid updating the shadow registers while writing new values in the preload registers. Then no update event occurs until the UDIS bit has been written to 0. However, the counter restarts from 0, as well as the counter of the prescaler (but the prescale rate does not change). In addition, if the URS bit (update request selection) in TIMx_CR1 register is set, setting the UG bit generates an update event UEV but without setting the UIF flag (thus no interrupt or DMA request is sent). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

When an update event occurs, all the registers are updated and the update flag (UIF bit in TIMx_SR register) is set (depending on the URS bit):

- The repetition counter is reloaded with the content of TIMx_RCR register.
- The auto-reload shadow register is updated with the preload value (TIMx_ARR).
- The buffer of the prescaler is reloaded with the preload value (content of the TIMx_PSC register).

The following figures show some examples of the counter behavior for different clock frequencies when TIMx_ARR = 0x36.

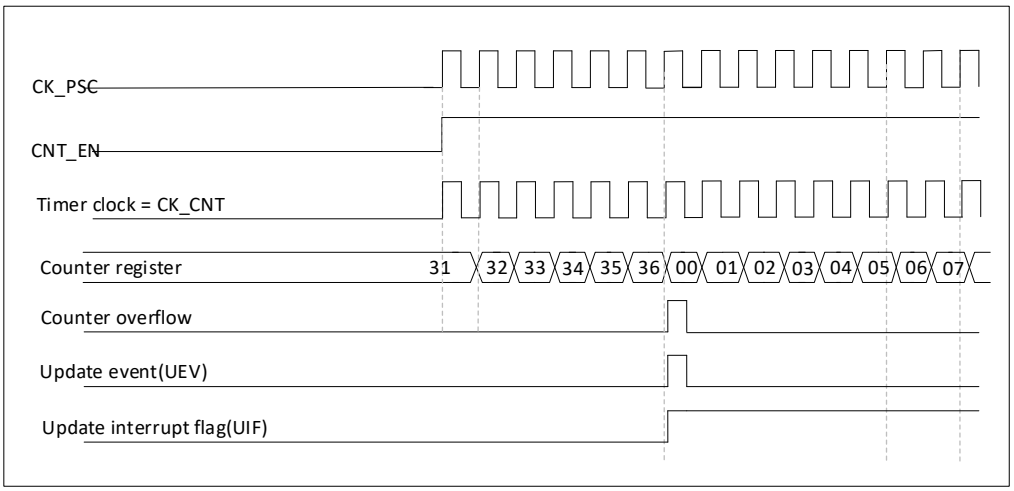

Figure 20-4 Counter timing diagram, internal clock divided by 1

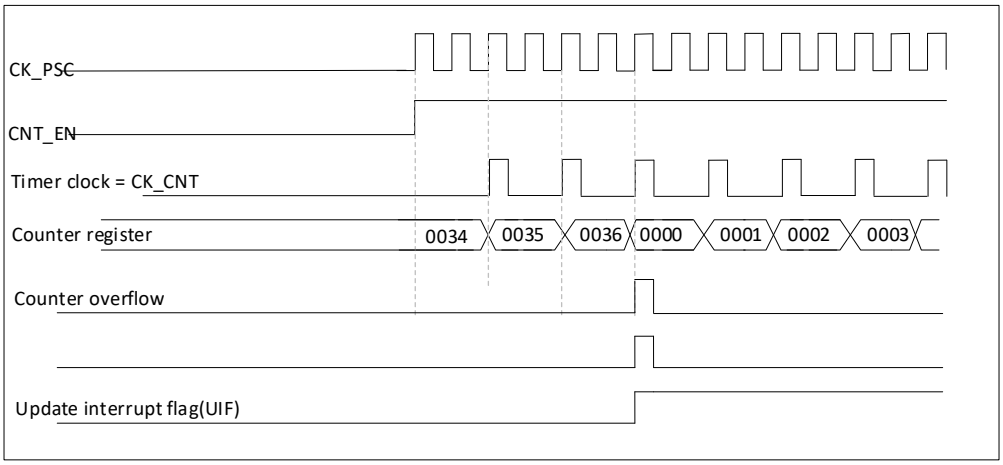

Figure 20-5 Counter timing diagram, internal clock divided by 2

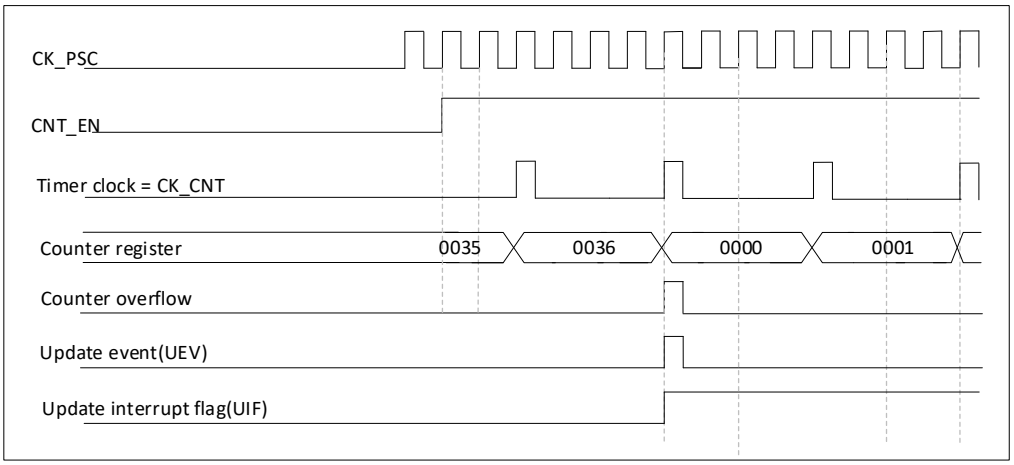

Figure 20-6 Counter timing diagram, internal clock divided by 4

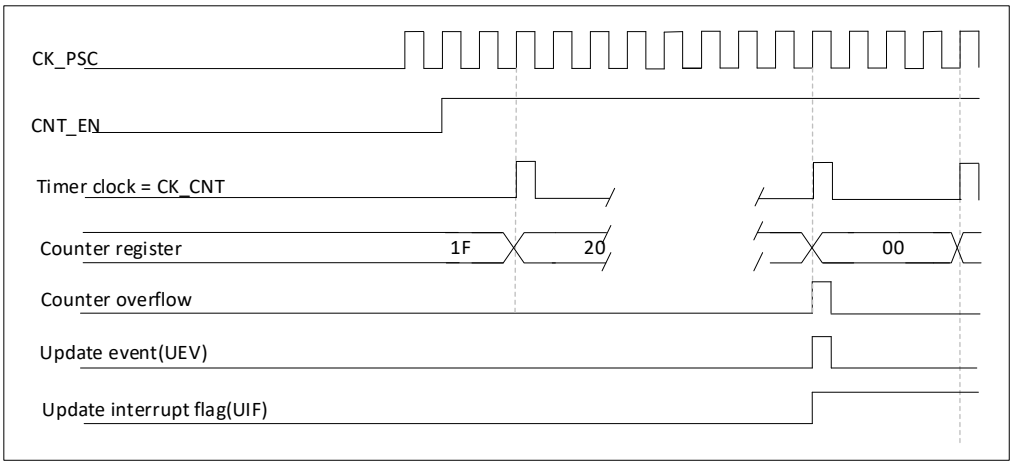

Figure 20-7 Counter timing diagram, internal clock divided by N

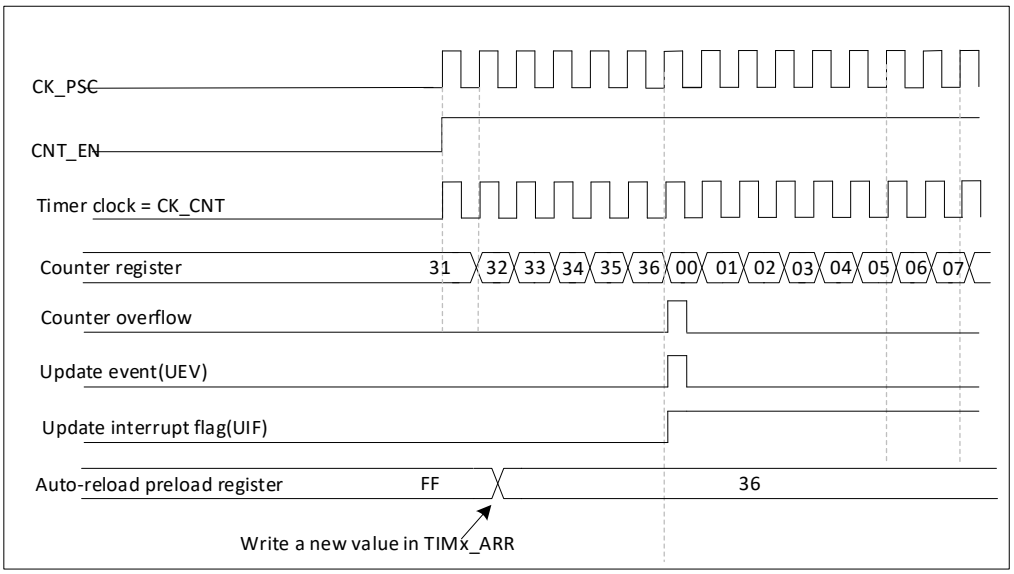

Figure 20-8 Counter timing diagram, update event when ARPE = 0 (TIMx_ARR not preloaded)

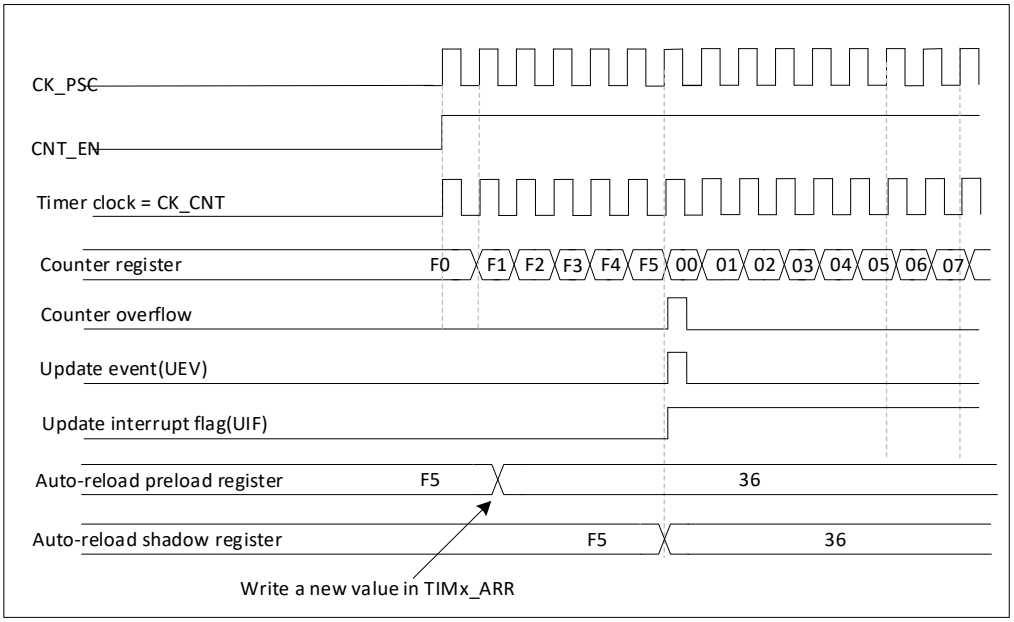

Figure 20-9 Counter timing diagram, update event when ARPE = 1 (TIMx_ARR preloaded)

20.2.3. Repetition counter

Time-base unit describes how the update event (UEV) is generated withrespect to the counter overflows/under-flows. It is actually generated only when the repetitioncounter has reached zero. This can be useful when gen-erating PWM signals.

This means that data are transferred from the preload registers to the shadow registers(TIMx_ARR auto-reload register, TIMx_PSC prescaler register, but also TIMx_CCRx capture/compare registers in compare mode) every N counter overflows or underflows, where N is the value in the TIMx_RCR repetition counter register.

The repetition counter is decremented at each counter overflow in upcounting mode.

The repetition counter is an auto-reload type, the repetition rate is maintained as defined by the TIMx_RCR register value. When the update event is generated by software (by setting the UG bit in TIMx_EGR register) or by hardware through the slave mode controller, it occurs immediately whatever the value of the repetition counter is and the repetition counter is reloaded with the content of the TIMx_RCR register.

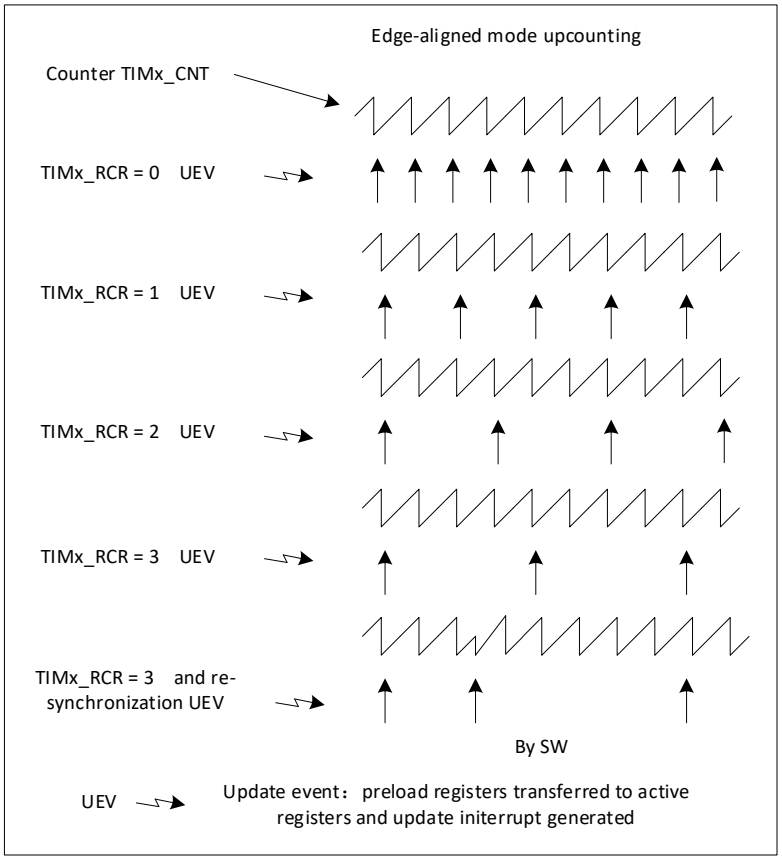

Figure 20-10 Update rate examples depending on mode and TIMx_RCR register settings

20.2.4. Clock sources

The counter clock can be provided Internal clock (CK_INT). the CEN and UG bits (in the TIMx_EGR register) are actual control bits and can be changed only by software (except UG which remains cleared automatically). As soon as the CEN bit is written to 1, the prescaler is clocked by the internal clock CK_INT.

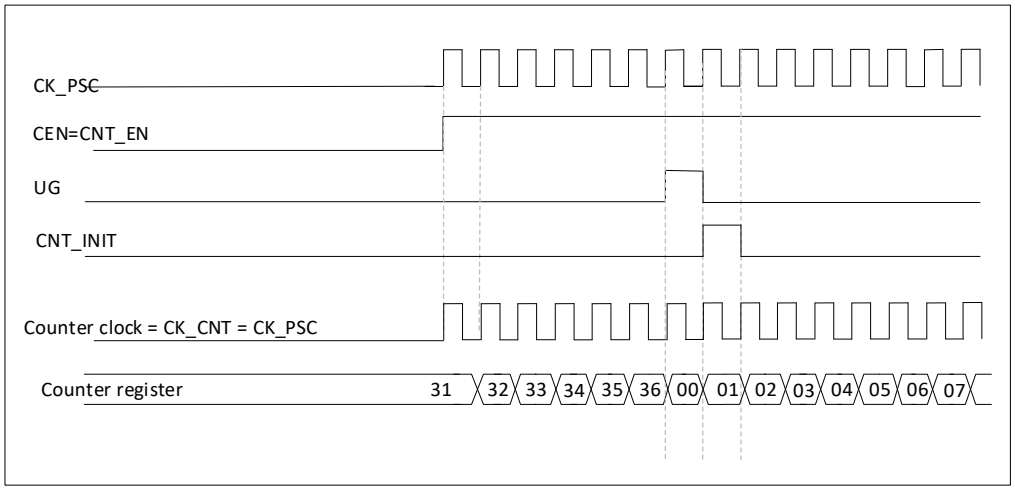

Figure 20-11 Control circuit in normal mode, internal clock divided by 1

20.2.5. Capture/compare channels

Each Capture/Compare channel is built around a capture/compare register (including a shadow register), a input stage for capture (with digital filter, multiplexing and prescaler) and an output stage (with comparator and output control).

The following figures give an overview of one Capture/Compare channel.

The input stage samples the corresponding TIx input to generate a filtered signal TIxF. Then, an edge detector with polarity selection generates a signal (TIxFPx) which can be used as trigger input by the slave mode controller or as the capture command. It is prescaled before the capture register (ICxPS).

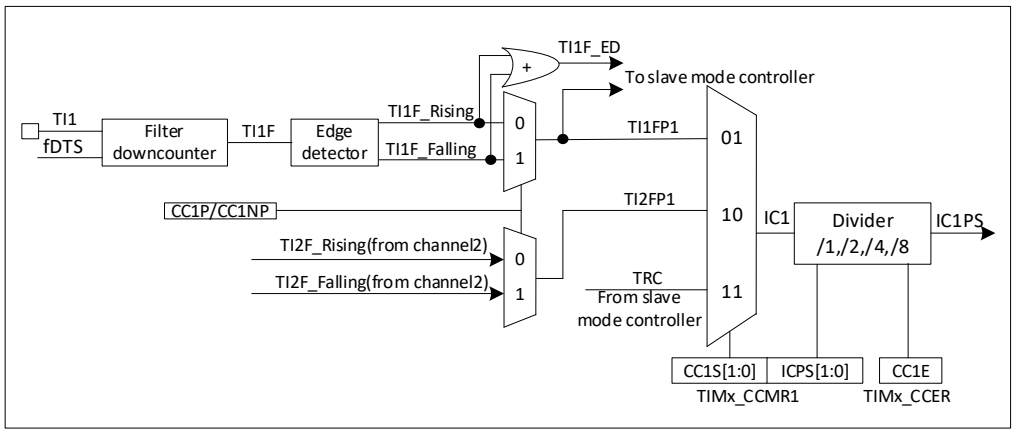

Figure 20-12 Capture/compare channel (example: channel 1 input stage)

The output stage generates an intermediate waveform which is then used for reference: OCxRef (active high). The polarity acts at the end of the chain.

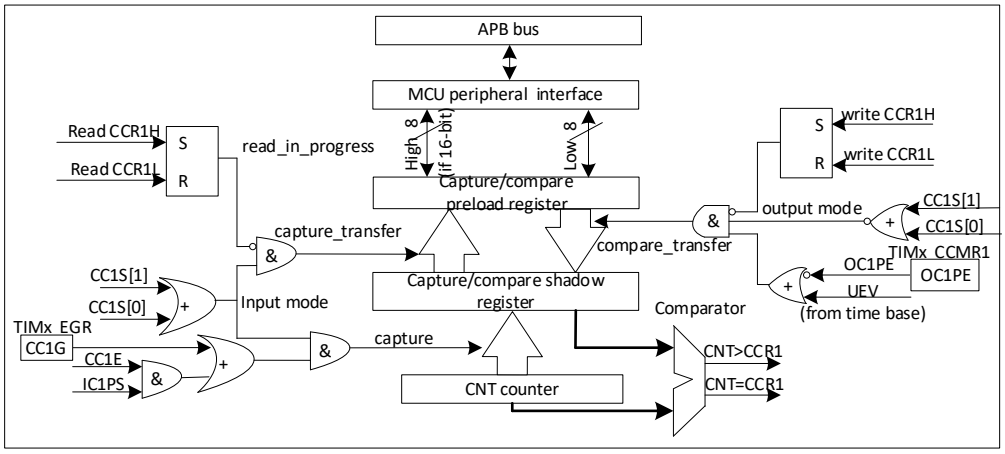

Figure 20-13 Capture/compare channel 1 main circuit

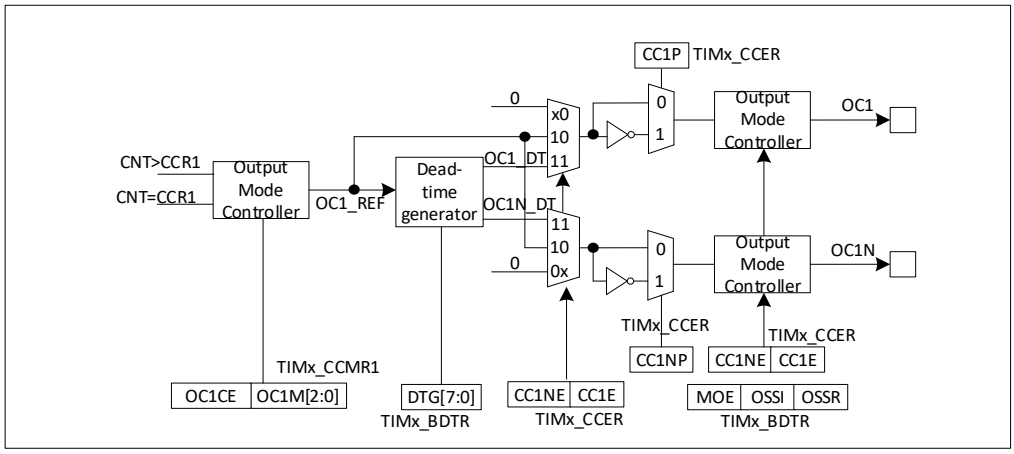

Figure 20-14 Output stage of capture/compare channel (channel 1)

The capture/compare block is made of one preload register and one shadow register. Write and read only access the preload register.

In capture mode, captures are actually done in the shadow register, which is copied into the preload register.
In compare mode, the content of the preload register is copied into the shadow register which is compared to the counter.

20.2.6. Input capture mode

In Input capture mode, the Capture/Compare Registers (TIMx_CCRx) are used to latch the value of the counter after a transition detected by the corresponding ICx signal. When a capture occurs, the corresponding CCXIF flag (TIMx_SR register) is set. If a capture occurs while the CCxIF flag was already high, then the over-capture flag CCxOF (TIMx_SR register) is set. CCxIF can be cleared by software by writing it to '0' or by reading the captured data stored in the TIMx_CCRx register. CCxOF is cleared when it is written with 0.
The following example shows how to capture the counter value in TIMx_CCR1 when TI1 input rises. To do this, use the following procedure:

- Select the active input: TIMx_CCR1 must be linked to the TI1 input, so write the CC1S bits to 01 in the TIMx_CCMR1 register. As soon as CC1S becomes different from 00, the channel is configured in input and the TIMx_CCR1 register becomes read-only.
- Program the appropriate input filter duration in relation with the signal connected to the timer (when the input is one of the TIx (ICxF bits in the TIMx_CCMRx register). Let's imagine that, when toggling, the input signal is not stable during at must 5 internal clock cycles. We must program a filter duration longer than these 5 clock cycles. We can validate a transition on TI1 when 8 consecutive samples with the new level have been detected (sampled at fDTS frequency). Then write IC1F bits to 0011 in the TIMx_CCMR1 register.
- Select the edge of the active transition on the TI1 channel by writing CC1P bit to 0 in the TIMx_CCER register (rising edge in this case).
- Program the input prescaler. In our example, we wish the capture to be performed at each valid transition, so the prescaler is disabled (write IC1PS bits to '00' in the TIMx_CCMR1 register).
- Enable capture from the counter into the capture register by setting the CC1E bit in the TIMx_CCER register.
- If needed, enable the related interrupt request by setting the CC1IE bit in the TIMx_DIER register, and/or the DMA request by setting the CC1DE bit in the TIMx_DIER register.

When an input capture occurs:

- The TIMx_CCR1 register gets the value of the counter on the active transition.
- CC1IF flag is set (interrupt flag). CC1OF is also set if at least two consecutive captures occurred whereas the flag was not cleared.
- An interrupt is generated depending on the CC1IE bit.
- A DMA request is generated depending on the CC1DE bit.

In order to handle the overcapture, it is recommended to read the data before the overcapture flag. This is to avoid missing an overcapture which could happen after reading the flag and before reading the data.
Note: IC interrupt and/or DMA requests can be generated by software by setting the corresponding CCxG bit in the TIMx_EGR register.

20.2.7. Forced output mode

In output mode (CCxS bits = 00 in the TIMx_CCMRx register), each output compare signal (OCxREF and then OCx) can be forced to active or inactive level directly by software, independently of any comparison between the output compare register and the counter.

To force an output compare signal (OCXREF/OCx) to its active level, one just needs to write 101 in the OCxM bits in the corresponding TIMx_CCMRx register. Thus OCXREF is forced high (OCxREF is always active high) and OCx get opposite value to CCxP polarity bit.

For example: CCxP = 0 (OCx active high) = > OCx is forced to high level.

The OCxREF signal can be forced low by writing the OCxM bits to 100 in the TIMx_CCMRx register.

Anyway, the comparison between the TIMx_CCRx shadow register and the counter is still performed and allows the flag to be set. Interrupt and DMA requests can be sent accordingly. This is described in the output compare mode section below.

20.2.8. Output compare mode

This function is used to control an output waveform or indicating when a period of time has elapsed.

When a match is found between the capture/compare register and the counter, the output compare function:

- Assigns the corresponding output pin to a programmable value defined by the output compare mode (OCxM bits in the TIMx_CCMRx register) and the output polarity (CCxP bit in the TIMx_CCER register). The output pin can keep its level (OCXM = 000), be set active (OCxM = 001), be set inactive (OCxM = 010) or can toggle (OCxM = 011) on match.
- Sets a flag in the interrupt status register (CCxIF bit in the TIMx_SR register).
- Generates an interrupt if the corresponding interrupt mask is set (CCXIE bit in the TIMx_DIER register).

The TIMx_CCRx registers can be programmed with or without preload registers using the OCxPE bit in the TIMx_CCMRx register.

In output compare mode, the update event UEV has no effect on OCxREF and OCx output. The timing resolution is one count of the counter. Output compare mode can also be used to output a single pulse (in One-pulse mode).

Procedure:

1. Select the counter clock (internal, external, prescaler).
2. Write the desired data in the TIMx_ARR and TIMx_CCRx registers.
3. Set the CCxIE bit if an interrupt request is to be generated.
4. Select the output mode. For example:

– Write OCxM = 011 to toggle OCx output pin when CNT matches CCRx

– Write OCxPE = 0 to disable preload register

– Write CCxP = 0 to select active high polarity

– Write CCxE = 1 to enable the output

5. Enable the counter by setting the CEN bit in the TIMx_CR1 register.

The TIMx_CCRx register can be updated at any time by software to control the output waveform, provided that the preload register is not enabled (OCxPE = ’0’, else TIMx_CCRx shadow register is updated only at the next update event UEV). An example is given in the following figure.

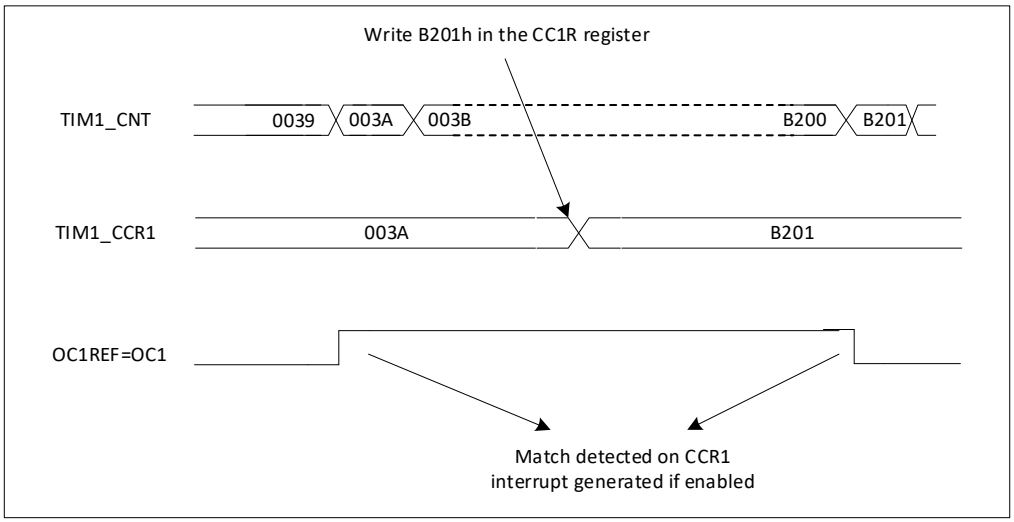

Figure 20-15 Output compare mode, toggle on OC1

20.2.9. PWM mode

Pulse Width Modulation mode allows to generate a signal with a frequency determined by the value of the TIMx_ARR register and a duty cycle determined by the value of the TIMx_CCRx register.

The PWM mode can be selected independently on each channel (one PWM per OCx output) by writing '110' (PWM mode 1) or '111' (PWM mode 2) in the OCxM bits in the TIMx_CCMRx register. The corresponding preload register must be enabled by setting the OCxPE bit in the TIMx_CCMRx register, and eventually the auto-reload preload register by setting the ARPE bit in the TIMx_CR1 register.

As the preload registers are transferred to the shadow registers only when an update event occurs, before starting the counter, all registers must be initialized by setting the UG bit in the TIMx_EGR register.

OCx polarity is software programmable using the CCxP bit in the TIMx_CCER register. It can be programmed as active high or active low. OCx output is enabled by CCxE bit (TIMx_CCER register).

In PWM mode (1 or 2), TIMx_CNT and TIMx_CCRx are always compared to determine whether TIMx_CCRx≤TIMx_CNT.

The timer is able to generate PWM in edge-aligned mode only since the counter is upcounting.

PWM edge-aligned mode

In the following example, we consider PWM mode 1. The reference PWM signal OCxREF is high as long as TIMx_CNT < TIMx_CCRx else it becomes low. If the compare value in TIMx_CCRx is greater than the auto-reload value (in TIMx_ARR) then OCxREF is held at'1'. If the compare value is 0 then OCxRef is held at '0'. The following figure shows some edgealigned PWM waveforms in an example where TIMx_ARR = 8.

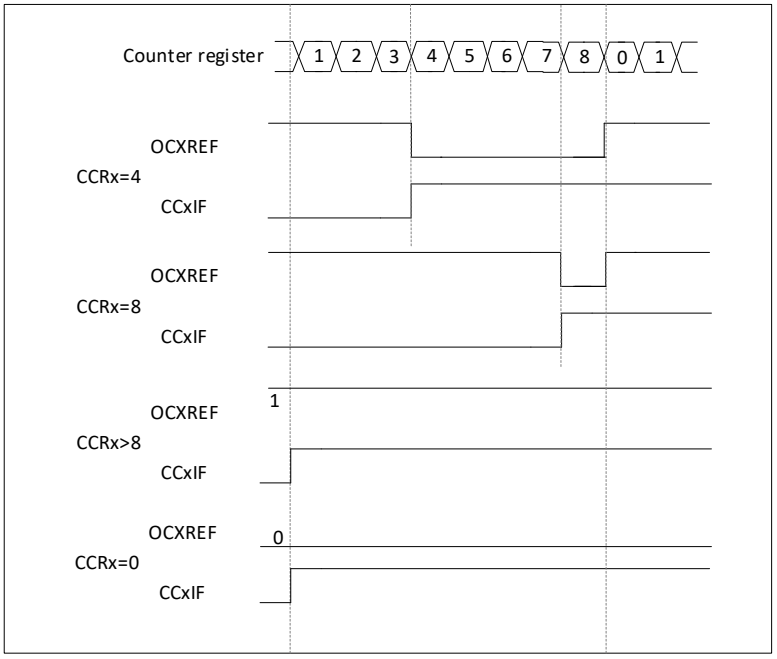

Figure 20-16 Edge-aligned PWM waveforms (ARR = 8)

20.2.10. Complementary outputs and dead-time insertion

The TIM16/17 general-purpose timers can output one complementary signal and manage the switching-off and switching-on of the outputs. This time is generally known as dead-time and it has to be adjusted depending on the devices that are connected to the outputs and their characteristics (intrinsic delays of levelshifters, delays due to power switches).

The polarity of the outputs (main output OCx or complementary OCxN) can be selected independently for each output. This is done by writing to the CCxP and CCxNP bits in the TIMx_CCER register.

The complementary signals OCx and OCxN are activated by a combination of several control bits: the CCxE and CCxNE bits in the TIMx_CCER register and the MOE, OISx, OISxN, OSSI and OSSR bits in the TIMx_BDTR and TIMx_CR2 registers. Refer to Table xx Output control bits for complementary OCx and OCxN channels with break feature for more details. In particular, the dead-time is activated when switching to the IDLE state (MOE falling down to 0).

Dead-time insertion is enabled by setting both CCxE and CCxNE bits, and the MOE bit if the break circuit is present. There is one 8-bit dead-time generator DTG[7:0] for each channel. From a reference waveform OCxREF, it generates 2 outputs OCx and OCxN. If OCx and OCxN are active high:

- The OCx output signal is the same as the reference signal except for the rising edge, which is delayed relative to the reference rising edge.
- The OCxN output signal is the opposite of the reference signal except for the rising edge, which is delayed relative to the reference falling edge.

If the delay is greater than the width of the active output (OCx or OCxN) then the corresponding pulse is not generated.

The following figures show the relationships between the output signals of the dead-time generator and the reference signal OCxREF. (we suppose CCxP = 0, CCxNP = 0, MOE = 1, CCxE = 1 and CCxNE = 1 in these examples).

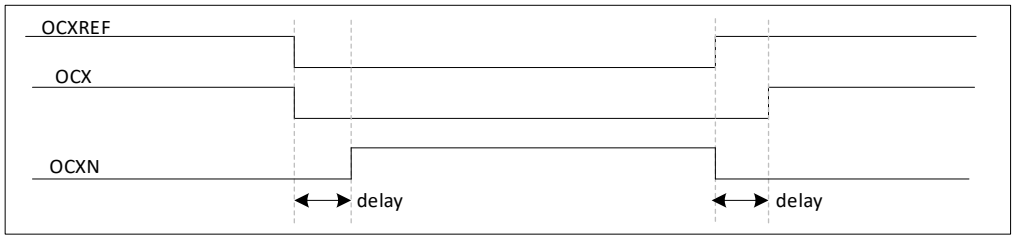

Figure 20-17 Complementary output with dead-time insertion

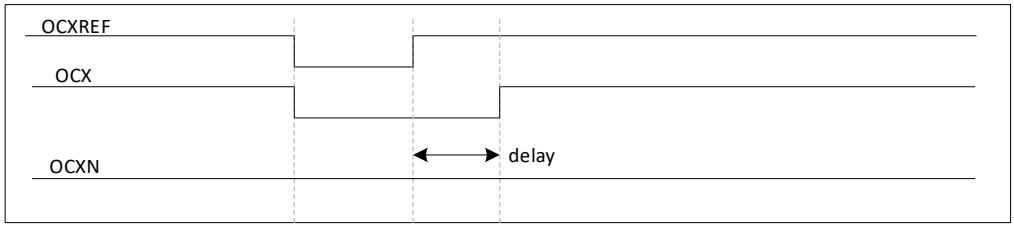

Figure 20-18 Dead-time waveforms with delay greater than the negative pulse

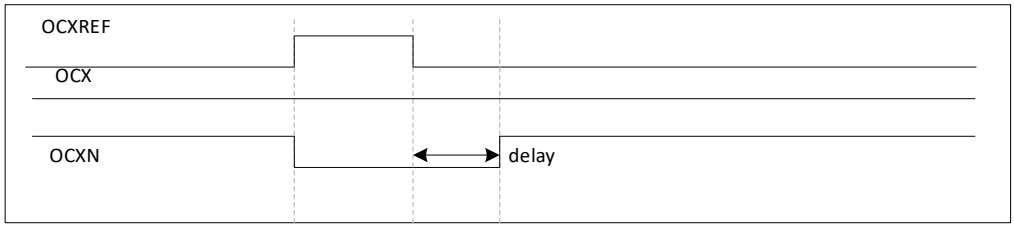

Figure 20-19 Dead-time waveforms with delay greater than the positive pulse

The dead-time delay is the same for each of the channels and is programmable with the DTG bits in the TIMx_BDTR register.

Re-directing OCxREF to OCx or OCxN

In output mode (forced, output compare or PWM), OCxREF can be re-directed to the OCx output or to OCxN output by configuring the CCxE and CCxNE bits in the TIMx_CCER register. This allows to send a specific waveform (such as PWM or static active level) on one output while the complementary remains at its inactive level. Other alternative possibilities are to have both outputs at inactive level or both outputs active and complementary with deadtime.

Note: When only OCxN is enabled (CCxE = 0, CCxNE = 1), it is not complemented and becomes active as soon as OCxREF is high. For example, if CCxNP = 0 then OCxN = OCxRef. On the other hand, when both OCx and OCxN are enabled (CCxE = CCxNE = 1) OCx becomes active when OCxREF is high whereas OCxN is complemented and becomes active when OCxREF is low.

20.2.11. Using the break function

When using the break function, the output enable signals and inactive levels are modified according to additional control bits (MOE, OSSI and OSSR bits in the TIMx_BDTR register, OISx and OISxN bits in the TIMx_CR2 register). In any case, the OCx and OCxN outputs cannot be set both to active level at a given time.

The source for break (BRK) channel can be an external source connected to the BKIN pin or one of the following internal sources:

- The core LOCKUP output
- The PVD output
- A clock failure event generated by the CSS detector

- The comparator output

When exiting from reset, the break circuit is disabled and the MOE bit is low. The break function can be enabled by setting the BKE bit in the TIMx_BDTR register. The break input polarity can be selected by configuring the BKP bit in the same register. BKE and BKP can be modified at the same time. When the BKE and BKP bits are written, a delay of 1 APB clock cycle is applied before the writing is effective. Consequently, it is necessary to wait 1 APB clock period to correctly read back the bit after the write operation.

Because MOE falling edge can be asynchronous, a resynchronization circuit has been inserted between the actual signal (acting on the outputs) and the synchronous control bit (accessed in the TIMx_BDTR register). It results in some delays between the asynchronous and the synchronous signals. In particular, if MOE is set to 1 whereas it was low, a delay must be inserted (dummy instruction) before reading it correctly. This is because the write acts on the asynchronous signal whereas the read reflects the synchronous signal.

When a break occurs (selected level on the break input):

- The MOE bit is cleared asynchronously, putting the outputs in inactive state, idle state or in reset state (selected by the OSSI bit). This feature functions even if the MCU oscillator is off.
- Each output channel is driven with the level programmed in the OISx bit in the TIMx_CR2 register as soon as MOE = 0. If OSSI = 0 then the timer releases the enable output else the enable output remains high.
- When complementary outputs are used:
  - The outputs are first put in reset state inactive state (depending on the polarity). This is done asynchronously so that it works even if no clock is provided to the timer.
  - If the timer clock is still present, then the dead-time generator is reactivated in order to drive the outputs with the level programmed in the OISx and OISxN bits after a dead-time. Even in this case, OCx and OCxN cannot be driven to their active level together. Note that because of the resynchronization on MOE, the dead-time duration is a bit longer than usual (around 2 ck_tim clock cycles).
  - If OSSI = 0 then the timer releases the enable outputs else the enable outputs remain or become high as soon as one of the CCxE or CCxNE bits is high.
- The break status flag (BIF bit in the TIMx_SR register) is set. An interrupt can be generated if the BIE bit in the TIMx_DIER register is set. A DMA request can be generated if the BDE bit in the TIMx_DIER register is set.
- If the AOE bit in the TIMx_BDTR register is set, the MOE bit is automatically set again at the next update event UEV. This can be used to perform a regulation, for instance. Else, MOE remains low until it is written with 1 again. In this case, it can be used for security and the break input can be connected to an alarm from power drivers, thermal sensors or any security components.

Note: The break inputs is acting on level. Thus, the MOE cannot be set while the break input is active (neither automatically nor by software). In the meantime, the status flag BIF cannot be cleared.

The break can be generated by the BRK input which has a programmable polarity and an enable bit BKE in the TIMx_BDTR Register.

There are two ways to generate break:

- BKR input with programmable polarity while enable BKE in TIMx_BDTR register.
- Set the BG bit in TIMx_EGR by software.

In addition to the break input and the output management, a write protection has been implemented inside the break circuit to safeguard the application. It allows to freeze the configuration of several parameters (dead-time duration, OCx/OCxN polarities and state when disabled, OCxM configurations, break enable and polarity). The protection can be selected among 3 levels with the LOCK bits in the TIMx_BDTR register. Refer to TIM1 break and dead-time register (TIM1x_BDTR). The LOCK bits can be written only once after an MCU reset.
The following figure shows an example of behavior of the outputs in response to a break.

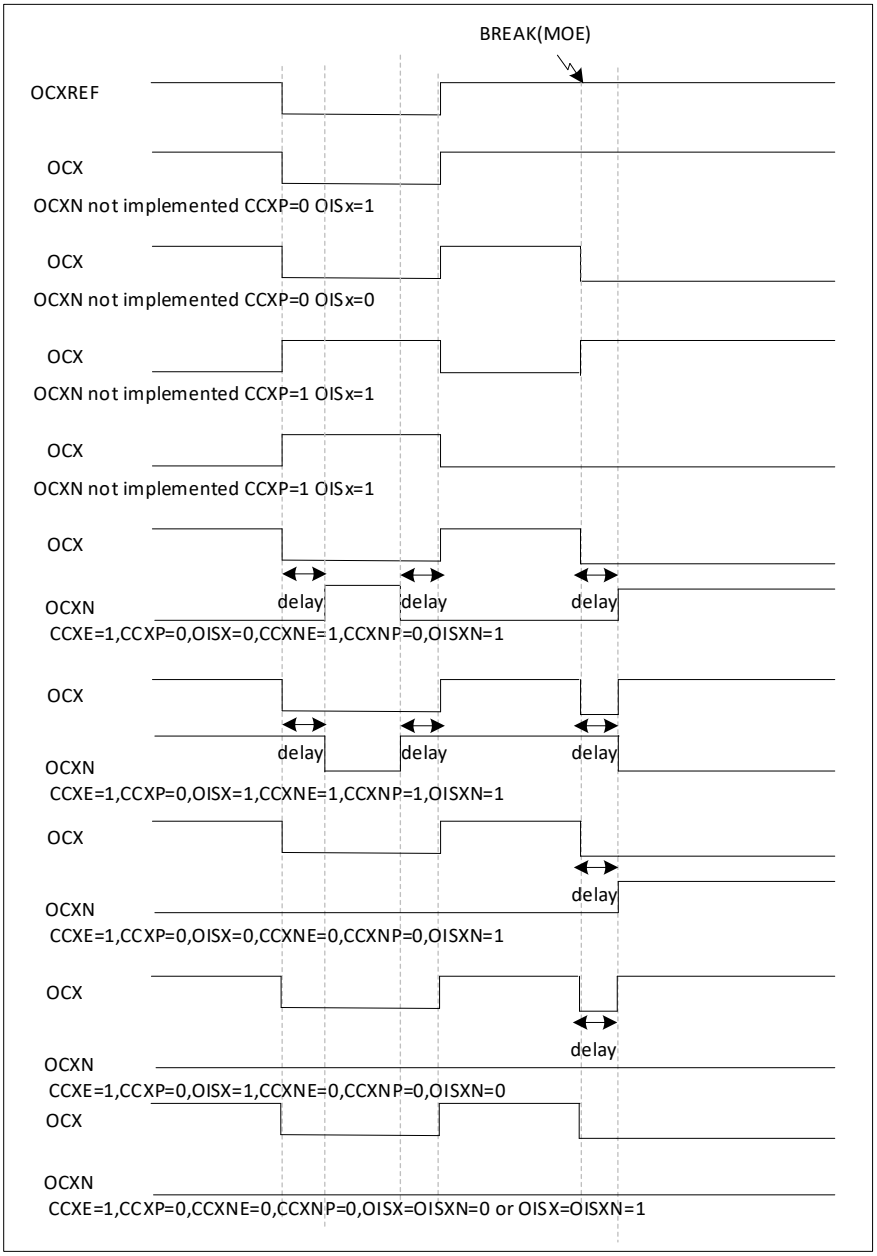

Figure 20-20 Output behavior in response to a break

20.2.12. One-pulse mode

One-pulse mode (OPM) is a particular case of the previous modes. It allows the counter to be started in response to a stimulus and to generate a pulse with a programmable length after a programmable delay.
Starting the counter can be controlled through the slave mode controller. Generating the waveform can be done in output compare mode or PWM mode. One-pulse mode is selected by setting the OPM bit in the TIMx_CR1 register. This makes the counter stop automatically at the next update event UEV.

A pulse can be correctly generated only if the compare value is different from the counter initial value. Before starting (when the timer is waiting for the trigger), the configuration must be:

- In upcounting: CNT < CCRx ≤ ARR (in particular, 0 < CCRx)
- In downcounting: CNT > CCRx

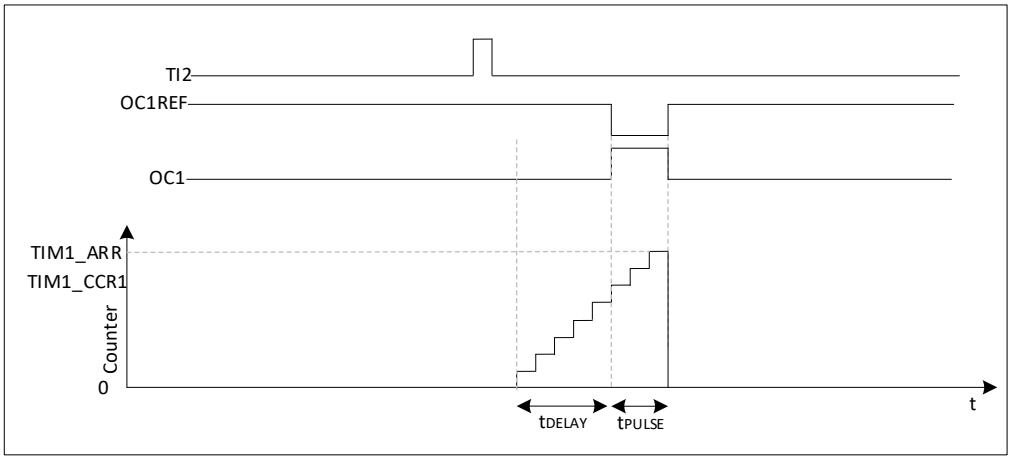

Figure 20-21 Example of One-pulse mode

Particular case: OCx fast enable

In One-pulse mode, the edge detection on TIx input set the CEN bit which enables the counter. Then the comparison between the counter and the compare value makes the output toggle. But several clock cycles are needed for these operations and it limits the minimum delay t_{DELAY} min we can get.

If one wants to output a waveform with the minimum delay, the OCxFE bit can be set in the TIMx_CCMRx register. Then OCxRef (and OCx) are forced in response to the stimulus, without taking in account the comparison. Its new level is the same as if a compare match had occurred. OCxFE acts only if the channel is configured in PWM1 or PWM2 mode.

20.3. TIM16/TIM17 registers

20.3.1. TIM16/17 control register 1 (TIMx_CR1)

Address offset: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	CKD[1:0]		ARPE	Res	Res	Res	OPM	URS	UDIS	CEN
						RW		RW				RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	0	Reserved, must be kept at reset value.
9:8	CKD[1:0]	RW	00	Clock division This bit-field indicates the division ratio between the timer clock (CK_INT) frequency and the dead-time and sampling clock (tDTS)used by the dead-time generators and the digital filters (TIx), 00: tDTS = tCK_INT 01: tDTS = 2*tCK_INT 10: tDTS = 4*tCK_INT 11: Reserved, do not program this value

7	ARPE	RW	0	Auto-reload preload enable 0: TIMx_ARR register is not buffered 1: TIMx_ARR register is buffered
6:4	Reserved	-	0	Reserved, must be kept at reset value.
3	OPM	RW	0	One pulse mode 0: Counter is not stopped at update event 1: Counter stops counting at the next update event (clearing the bit CEN)
2	URS	RW	0	Update request source This bit is set and cleared by software to select the UEV event sources. 0: Any of the following events generate an update interrupt or DMA request if enabled. These events can be: – Counter overflow/underflow – Setting the UG bit – Update generation through the slave mode controller 1: Only counter overflow/underflow generates an update interrupt or DMA request if enabled.
1	UDIS	RW	0	Update disable This bit is set and cleared by software to enable/disable UEV event generation. 0: UEV enabled. The Update (UEV) event is generated by one of the following events: – Counter overflow/underflow – Setting the UG bit – Update generation through the slave mode controller Buffered registers are then loaded with their preload values. 1: UEV disabled. The Update event is not generated, shadow registers keep their value (ARR, PSC, CCRx). However the counter and the prescaler are reinitialized if the UG bit is set or if a hardware reset is received from the slave mode controller.
0	CEN	RW	0	Counter enable 0: Counter disabled 1: Counter enabled Note: External clock, gated mode and encoder mode can only work after the CEN bit is set by software. Trigger mode can automatically set the CEN bit by hardware.

20.3.2. TIM16/17 control register 2 (TIM16/17_CR2)

Address offset: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	OIS1N	OIS1	Res	Res	Res	Res	CCDS	Res	Res	CCPC
						RW	RW					RW			RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	0	Reserved, must be kept at reset value.
9	OIS1N	RW	0	Output Idle state 1 (OC1N output) 0: OC1N = 0 after a dead-time when MOE = 0 1: OC1N = 1 after a dead-time when MOE = 0 Note: This bit can not be modified as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in TIMx_BKR register).
8	OIS1	RW	0	Output Idle state 1 (OC1 output) 0: OC1 = 0 (after a dead-time if OC1N is implemented) when MOE = 0 1: OC1 = 1 (after a dead-time if OC1N is implemented) when MOE = 0 Note: This bit can not be modified as long as LOCK level 1, 2 or 3 has been programmed

				(LOCK bits in TIMx_BKR register).
7:4	Reserved	-	0	Reserved, must be kept at reset value.
3	CCDS	RW	0	Capture/compare DMA selection 0: CCx DMA request sent when CCx event occurs 1: CCx DMA requests sent when update event occurs
2:1	Res	-	0	Reserved, must be kept at reset value.
0	CCPC	RW	0	Capture/compare preloaded control 0: CCxE, CCxNE and OCxM bits are not preloaded 1: CCxE, CCxNE and OCxM bits are preloaded, after having been written, they are updated only when COMG bit is set. Note: This bit acts only on channels that have a complementary output.

20.3.3. TIM16/17 DMA/interrupt enable register (TIM16/17_DIER)

Address offset: 0x0C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
						CC1DE	UDE	BIE	Res	COMIE	Res	Res	Res	CC1IE	UIE
						RW	RW	RW		RW				RW	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	0	Reserved, must be kept at reset value.
9	CC1DE	RW	0	CC1DE: capture/compare 1 DMA request enable 0: capture/compare 1 DMA request disabled. 1: capture/compare 1 DMA request enabled.
8	UDE	RW	0	UDE: Update DMA request enable 0: Update DMA request disabled. 1: Update DMA request enabled.
7	BIE	RW	0	BIE: break interrupt enable 0: break interrupt disabled. 1: break interrupt enabled.
6	Reserved	-	0	Reserved, must be kept at reset value.
5	COMIE	RW	0	COMIE: COM interrupt enable 0: COM interrupt disabled. 1: COM interrupt enabled.
4:2	Reserved	-	0	Reserved, must be kept at reset value.
1	CC1IE	RW	0	CC1IE: capture/compare 1 interrupt enable 0: capture/compare 1 interrupt disabled. 1: capture/compare 1 interrupt enabled.
0	UIE	RW	0	UIE: Update interrupt enable 0: Update interrupt disabled. 1: Update interrupt enabled.

20.3.4. TIM16/17 status register (TIM16/17_SR)

Address offset: 0x010

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	CC1OF	Res	BIF	Res	COMIF	Res	Res	Res	CC1F	UIF
						RC_W0		RC_W0		RC_W0				RC_W0	RC_W0

Bit	Name	R/W	Reset Value	Function

31:10	Reserved	-	0	Reserved, must be kept at reset value.
9	CC1OF	Rc_w0	0	capture/compare 1 over capture flag This flag can be set by hardware only when the corresponding channel is configured for input capture. Writing 0 clears this bit. 0: No overcapture is generated, 1: When CC1IF is set, the counter value has been captured into the TIMx_CCR1 register.
8	Reserved	Rc_w0	0	Reserved, must be kept at reset value.
7	BIF	Rc_w0	0	break Interrupt flag This bit is set by hardware once the break input is vaild. This bit can be cleared by software if the break input is invaild. 0: No break event is generated, 1: A valid level is detected on the break input.
6	Reserved	-	0	Reserved, must be kept at reset value.
5	COMIF	Rc_w0	0	COM interrupt flag This bit is set by hardware once a COM event is generated (when CcxE, CcxNE, OCxM have been updated). It is cleared by software. 0: No COM event is generated, 1: COM interrupt waiting for response
4:2	Reserved	-	0	Reserved, must be kept at reset value.
1	CC1IF	Rc_w0	0	Capture/Compare 1 Interrupt Flag If channel CC1 is configured in output mode: This bit is set by hardware when the counter value matches the compare value, except in center-align mode (refer to the CMS bit in the TIM1_CR1 register). It is cleared by software. 0: no match occurs, 1: The value of TIMx_CNT matches the value of TIMx_CCR1. If channel CC1 is configured in input mode: This bit is set by hardware when a capture event occurs, it is cleared by software or cleared by reading TIMx_CCR1. 0: No input capture is generated, 1: Input capture occurred and the counter value has been loaded into TIMx_CCR1 (edge detected on IC1 with the same polarity as selected).
0	UIF	Rc_w0	0	Update interrupt flag This bit is set by hardware on an update event. It is cleared by software. 0: No update occurred. 1: Update interrupt pending. This bit is set by hardware when the registers are updated: –At overflow or underflow regarding the repetition counter value and if UDIS = 0 in the TIMx_CR1 register. –When CNT is reinitialized by software using the UG bit in the TIMx_EGR register, if URS = 0 and UDIS = 0 in the TIMx_CR1 register.

20.3.5. TIM16/17 event generation register (TIM16/17_EGR)

Address offset: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	BG	Res	COMG	Res	Res	Res	CC1G	UG
								W		W				W	W

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	0	Reserved, must be kept at reset value.
7	BG	W	0	generate break event This bit is set by software to generate a break event and is automatically cleared by hardware.

				0: no action, 1: Generate a break event. At this time, MOE = 0, BIF = 1, if the corresponding interrupt and DMA are enabled, the corresponding interrupt and DMA will be generated.
6	Reserved	-	0	Reserved, must be kept at reset value.
5	COMG	W	0	Capture/compare events, generate control update This bit is set by software and automatically cleared by hardware. 0: no action, 1: When CCPC = 1, the CcxE, CcxNE, OCxM bits are allowed to be updated. Note: This bit is only valid for channels with complementary output.
4:2	Reserved	-	0	Reserved, must be kept at reset value.
1	CC1G	W	0	Generate capture/compare 1 event This bit is set by software to generate a capture/compare event and is automatically cleared by hardware. 0: no action, 1: Generate a capture/compare event on channel CC1: If channel CC1 is configured as an output: Set CC1IF = 1, if the corresponding interrupt and DMA are enabled, the corresponding interrupt and DMA will be generated. If channel CC1 is configured as an input: The current counter value is captured to the TIMx_CCR1 register, and CC1IF = 1 is set. If the corresponding interrupt and DMA are enabled, the corresponding interrupt and DMA will be generated. If CC1IF is already 1, set CC1OF = 1.
0	UG	W	0	Update generation This bit can be set by software, it is automatically cleared by hardware. 0: No action. 1: Re-initializes the timer counter and generates an update of the registers. Note that the prescaler counter is cleared too (but the prescaler ratio is not affected). If in center-align mode or DIR = 0 (upcounter), the counter will be cleared to 0, if DIR = 1 (downcounter), the counter will take the value of TIMx_ARR.

20.3.6. TIM16/17 capture/compare mode register 1 (TIM16/17_CCMR1)

Address offset: 0x18

Reset value: 0x0000 0000

Output compare mode:

31	**30**	**29**	**28**	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	Res	OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	
Res	Res	Res	Res	Res	Res	Res	Res	IC1F[3:0]				IC1PSC[1:0]			
								RW	RW	RW	RW	RW	RW	RW	

Output compare mode

Bit	Name	R/W	Reset Value	Function
31:7	Reserved	-		Reserved, must be kept at reset value.
6:4	OC1M[2:0]	RW	00	Output Compare 1 mode These bits define the behavior of the output reference signal OC1REF from which OC1 and OC1N are derived. OC1REF is active high whereas OC1 and OC1N active level depends on CC1P and CC1NP bits. 000: Frozen - The comparison between the output compare register TIMx_CCR1 and the counter TIMx_CNT has no effect on the outputs.

				001: Set channel 1 to active level on match. OC1REF signal is forced high when the counter TIMx_CNT matches the capture/compare register 1 (TIMx_CCR1). 010: Set channel 1 to inactive level on match. OC1REF signal is forced low when the counter TIMx_CNT matches the capture/compare register 1 (TIMx_CCR1). 011: Toggle - OC1REF toggles when TIMx_CNT = TIMx_CCR1. 100: Force inactive level - OC1REF is forced low. 101: Force active level - OC1REF is forced high. 110: PWM mode 1 - In upcounting, channel 1 is active as long as TIMx_CNT < TIMx_CCR1 else inactive. In downcounting, channel 1 is inactive (OC1REF = '0') as long as TIMx_CNT > TIMx_CCR1 else active (OC1REF = '1'). 111: PWM mode 2 - In upcounting, channel 1 is inactive as long as TIMx_CNT < TIMx_CCR1 else active. In downcounting, channel 1 is active as long as TIMx_CNT > TIMx_CCR1 else inactive. Note: These bits can not be modified as long as LOCK level 3 has been programmed (LOCK bits in TIMx_BDTR register) and CC1S = '00' (the channel is configured in output). Note: In PWM mode 1 or 2, the OCREF level changes only when the result of the comparison changes or when the output compare mode switches from "frozen" mode to "PWM" mode.
3	OC1PE	RW	0	Output Compare 1 Preload Enable 0: The preload function of the TIM1_CCR1 register is disabled, the TIM1_CCR1 register can be written at any time, and the new value takes effect immediately. 1: Enable the preload function of the TIM1_CCR1 register. The read and write operations are only performed on the preload register. The preload value of TIM1_CCR1 is loaded into the current register when the update event arrives. Note 1: This bit cannot be modified once the LOCK level is set to 3 (LOCK bit in the TIMx_BDTR register) and CC1S = 00 (the channel is configured as an output). Note 2: Only in single pulse mode, PWM mode can be used without confirming the preload register, otheRWise its action is undefined.
2	OC1FE	RW	0	Output Compare 1 Fast Enable This bit is used to speed up the CC output's response to trigger input events. 0: CC1 operates normally according to the value of the counter and CCR1, even if the flip-flop is on. When the flip-flop input has an active edge, the minimum delay to activate the CC1 output is 5 clock cycles. 1: The active edge input to the flip-flop acts as if a compare match had occurred. Therefore, OC is set to the comparison level regardless of the comparison result. The delay between the active edge of the sampling flip-flop and the CC1 output is shortened to 3 clock cycles. OC1FE only works when the channel is configured in PWM1 or PWM2 mode.
1:0	CC1S[1:0]	RW	00	Capture/Compare 1 Select. These 2 bits define the direction of the channel (input/output), and the selection of the input pins: 00: CC1 channel is configured as output, 01: CC1 channel is configured as input, IC1 is mapped on TI1, 10: Reserved 11: Reserved. Note: CC1S is writable only when the channel is off (CC1E = 0 in TIM16/17_CCER register).

Input Capture mode:

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-		Reserved, must be kept at reset value.

7:4	IC1F[3:0]	RW	0000	Input capture 1 filter This bit-field defines the frequency used to sample TI1 input and the length of the digital filter applied to TI1. The digital filter is made of an event counter in which N events are needed to validate a transition on the output: 0000: No filter, sampling is done at fDTS 0001: fSAMPLING = fCK_INT, N = 2 0010: fSAMPLING = fCK_INT, N = 4 0011: fSAMPLING = fCK_INT, N = 8 0100: fSAMPLING = fDTS / 2, N = 6 0101: fSAMPLING = fDTS / 2, N = 8 0110: fSAMPLING = fDTS / 4, N = 6 0111: fSAMPLING = fDTS / 4, N = 8 1000: fSAMPLING = fDTS / 8, N = 6 1001: fSAMPLING = fDTS / 8, N = 8 1010: fSAMPLING = fDTS / 16, N = 5 1011: fSAMPLING = fDTS / 16, N = 6 1100: fSAMPLING = fDTS / 16, N = 8 1101: fSAMPLING = fDTS / 32, N = 5 1110: fSAMPLING = fDTS / 32, N = 6 1111: fSAMPLING = fDTS / 32, N = 8
3:2	IC1PSC[1:0]	RW	00	Input capture 1 prescaler This bit-field defines the ratio of the prescaler acting on CC1 input (IC1). The prescaler is reset as soon as CC1E = ’0’ (TIM1_CCER register). 00: no prescaler, capture is done each time an edge is detected on the capture input 01: capture is done once every 2 events 10: capture is done once every 4 events 11: capture is done once every 8 events
1:0	CC1S[1:0]	RW	00	Capture/Compare 1 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC1 channel is configured as output 01: CC1 channel is configured as input, IC1 is mapped on TI1 Other: Reserved Note: CC1S bits are writable only when the channel is OFF (CC1E = 0 in TIM16/17_CCER).

Table output control bits for complementary OCx and OCxN channels with break feature

Control bits					Output states	
MOE	**OSSI**	**OSSR**	**CcxE**	**CcxNE**	**OCx output state**	**OCxN output state**
1	X	0	0	0	Output disabled(Not driven by the timer) OCx = 0, OCx_EN = 0	Output disabled(Not driven by the timer) OCxN = 0, OCxN_EN = 0
		0	0	1	Output disabled(Not driven by the timer) OCx = 0, OCx_EN = 0	OCxREF + Polarity OCxN = OCxREF xor CCxNP OCxN_EN = 1
		0	1	0	OCxREF + Polarity OCx = OCxREF xor CCxP OCx_EN = 1	Output disabled(Not driven by the timer)， OCxN = 0, OCxN_EN = 0
		0	1	1		
		1	0	0		
		1	0	1		
		1	1	0		
		1	1	1	OCREF+Polarity+dead time, OCx_EN = 1	OCREF Complementary value (Not OCREF)
0	0	X	0	0		
	0		0	1		
	0		1	0		
	0		1	1		
	1		0			
	1		0	1		
	1		1	0		
	1		1	1		

20.3.7. TIM16/17 capture/compare enable register (TIM16/17_CCER)

Address offset: 0x20

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	CC1NP	CC1NE	CC1P	CC1E
												RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:4	Reserved	-	0	Reserved, must be kept at reset value.
3	CC1NP	RW	0	Input/Capture 1 complementary output polarity 0: OC1N active high 1: OC1N active low Note: This bit is not writable as soon as LOCK level 2 or 3 has been programmed (LOCK bits in TIMx_BDTR register) and CC1S = ”00” (the channel is configured in output).
2	CC1NE	RW	0	Input/Capture/ 1 complementary output enable 0: Off - OC1N is not active. OC1N level is then function of MOE, OSSI, OSSR, OIS1, OIS1N and CC1E bits. 1: On - OC1N signal is output on the corresponding output pin depending on MOE, OSSI, OSSR, OIS1, OIS1N and CC1E bits.
1	CC1P	RW	0	Input/Capture 1 Output Polarity The CC1 channel is configured as an output: 0: OC1 active high 1: OC1 active low CC1 channel configured as input: The two bits of CC1NP/CC1P select whether the polarity signal of TI1FP1 or TI2FP1 is used as the trigger or capture signal. 00: Not inverted/rising edge: Capture occurs on the rising edge of TixFP1 (capture, reset trigger, external clock or trigger mode), TixFP1 is not inverted (gate trigger mode, code mode). 01: Inversion/Falling Edge: Not Inverted/Rising Edge: Capture occurs on the falling edge of TixFP1 (capture, reset trigger, external clock or trigger mode), TixFP1 is inverted (gate trigger mode, code mode). 10: Reserved, invalid configuration. 11: No reverse, double edge. Note: This bit cannot be modified once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 3 or 2 and CC1S = 00 (channel configured as output).
0	CC1E	RW	0	Input/Capture 1 Output Enable The CC1 channel is configured as an output: 0: Off - OC1 output is disabled, so the output level of OC1 depends on the value of the MOE, OSSI, OSSR, OIS1, OIS1N, CC1NE bits. 1: On - The OC1 signal is output to the corresponding output pin, and its output level depends on the value of the MOE, OSSI, OSSR, OIS1, OIS1N, CC1NE bits. The CC1 channel is configured as an input: This bit determines whether the value of the counter can capture the TIMx_CCR1 register. 0: Capture disabled 1: Capture enable

20.3.8. TIM16/17 counter (TIM16/17_CNT)

Address offset: 0x24

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CNT[15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			Reserved, must be kept at reset value.
15:0	CNT[15:0]	RW	0	Counter value

20.3.9. TIM16/17 prescaler (TIM16/17_PSC)

Address offset: 0x28

Reset value: 0x0000 0000

31	**30**	**29**	**28**	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PSC[15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			Reserved, must be kept at reset value.
15:0	PSC[15:0]	RW	0	Prescaler value The counter clock frequency (CK_CNT) is equal to fCK_PSC / (PSC[15:0] + 1). PSC contains the value to be loaded in the active pre-scaler register at each update event (including when the counter is cleared through UG bit of TIMx_EGR register or through trigger controller when configured in “reset mode”).

20.3.10. TIM16/17 auto-reload register (TIM16/17_ARR)

Address offset: 0x2c

Reset value: 0x0000 FFFF

31	**30**	**29**	**28**	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ARR[15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			Reserved, must be kept at reset value.
15:0	ARR[15:0]	RW	0xFFFF	Auto-reload value ARR is the value to be loaded in the actual auto-reload register. The counter is blocked while the auto-reload value is null.

20.3.11. TIM16/17 repetition counter register (TIM16/17_RCR)

Address offset: 0x30

Reset value: 0x0000 0000

31	**30**	**29**	**28**	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	REP[7:0]							
-	-	-	-	-	-	-	-	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:8	Reserved			Reserved, must be kept at reset value.
7:0	REP[7:0]	RW	0	Repetition counter value These bits allow the user to set-up the update rate of the compare registers (i.e. periodic transfers from preload to active registers) when preload registers are enable, as well as the update interrupt generation rate, if this interrupt is enable. Each time the REP_CNT related downcounter reaches zero, an update event is generated and it restarts counting from REP value. As REP_CNT is reloaded with REP value only at the repetition update event U_RC, any write to the TIMx_RCR register is not taken in account until the next repetition update event. It means in PWM mode (REP+1) corresponds to the number of PWM periods in edgealigned mode, the number of PWM half periods in center-aligned mode.

20.3.12. TIM16/17 capture/compare register 1 (TIM16/17_CCR1)

Address offset: 0x34

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR1[15:0]															
RW/RO															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			Reserved, must be kept at reset value.
15:0	CCR1[15:0]	RW	0	Capture/Compare 1 value Condition: channel CC1 is configured as output CCR1 is the value to be loaded in the actual capture/compare 1 register (preload value). It is loaded permanently if the preload feature is not selected in the TIMx_CCMR1 register (bit OC1PE). Else the preload value is copied in the active capture/compare 1 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC1 output. Condition: channel CC1 is configured as input: CCR1 is the counter value transferred by the last input capture 1 event (IC1).

20.3.13. TIM16/17 break and dead-time register (TIM16/17_BDTR)

Address offset: 0x44

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOE	AOE	BKP	BKE	OSSR	OSSI	LOCK[1:0]		DTG[7:0]							
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	RW	0	Reserved, must be kept at reset value.
15	MOE	RW	0	Main output enable This bit is cleared asynchronously by hardware as soon as the break input is active. It is set by software or automatically depending on the AOE bit. It is acting only on the channels which are configured in output.

				0: OC and OCN outputs are disabled or forced to idle state 1: OC and OCN outputs are enabled if their respective enable bits are set (CCxE, CCxNE in TIMx_CCER register) See OC/OCN enable description for more details (TIMx capture/compare enable register (TIMx_CCER)).
14	AOE	RW	0	Automatic output enable 0: MOE can be set only by software 1: MOE can be set by software or automatically at the next update event (if the break input is not be active) Note: This bit can not be modified as long as LOCK level 1 has been programmed (LOCK bits in TIMx_BDTR register).
13	BKP	RW	0	Break polarity 0: Break input BRK is active low 1: Break input BRK is active high Note: This bit can not be modified as long as LOCK level 1 has been programmed (LOCK bits in TIMx_BDTR register).
12	BKE	RW	0	Break enable 0: Break inputs (BRK and BRK_ACTH) disabled 1, Break inputs (BRK and BRK_ACTH) enabled Note: This bit cannot be modified when LOCK level 1 has been programmed (LOCK bits in TIMx_BDTR register).
11	OSSR	RW	0	Off-state selection for Run mode This bit is used when MOE = 1 on channels having a complementary output which are configured as outputs. OSSR is not implemented if no complementary output is implemented in the timer. See OC/OCN enable description for more details. 0: When inactive, OC/OCN outputs are disabled (OC/OCN enable output signal = 0) 1: When inactive, OC/OCN outputs are enabled with their inactive level as soon as CCxE = 1 or CCxNE = 1. Then, OC/OCN enable output signal = 1 Note: This bit can not be modified as soon as the LOCK level 2 has been programmed (LOCK bits in TIMx_BDTR register).
10	OSSI	RW	0	Off-state selection for Idle mode This bit is used when MOE = 0 on channels configured as outputs. See OC/OCN enable description for more details. 0: When inactive, OC/OCN outputs are disabled (OC/OCN enable output signal = 0) 1: When inactive, OC/OCN outputs are forced first with their idle level as soon as CCxE = 1 or CCxNE = 1. OC/OCN enable output signal = 1. Note: This bit can not be modified as soon as the LOCK level 2 has been programmed (LOCK bits in TIMx_BDTR register).
9:8	LOCK[1:0]	RW	00	Lock configuration These bits offer a write protection against software errors. 00: LOCK OFF - No bit is write protected 01: LOCK Level 1 = DTG bits in TIMx_BDTR register, OISx and OISxN bits in TIMx_CR2 register and BKE/BKP/AOE bits in TIMx_BDTR register can no longer be written. 10: LOCK Level 2 = LOCK Level 1 + CC Polarity bits (CCxP/CCxNP bits in TIMx_CCER register, as long as the related channel is configured in output through the CCxS bits) as well as OSSR and OSSI bits can no longer be written. 11: LOCK Level 3 = LOCK Level 2 + CC Control bits (OCxM and OCxPE bits in TIMx_CCMRx registers, as long as the related channel is configured in output through the CCxS bits) can no longer be written. Note: The LOCK bits can be written only once after the reset. Once the TIMx_BDTR register has been written, their content is frozen until the next reset.
7:0	DTG[7:0]	RW	0000 0000	Dead-time generator setup

				This bit-field defines the duration of the dead-time inserted between the complementary outputs. DT correspond to this duration. DTG[7:5] = 0xx = > DT = DTG[7:0]x tdtg with tdtg = tDTS DTG[7:5] = 10x = > DT = (64+DTG[5:0])xtdtg with Tdtg = 2xtDTS DTG[7:5] = 110 = > DT = (32+DTG[4:0])xtdtg with Tdtg = 8xtDTS DTG[7:5] = 111 = > DT = (32+DTG[4:0])xtdtg with Tdtg = 16xtDTS Example if TDTS = 125ns (8MHz), dead-time possible values are: 0 to 15875 ns by 125 ns steps, 16 µs to 31750 ns by 250 ns steps, 32 µs to 63 µs by 1 µs steps, 64 µs to 126 µs by 2 µs steps Note: This bit-field can not be modified as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in TIMx_BDTR register).

20.3.14. TIM16/17 DMA control register (TIM16/17_DCR)

Address offset: 0x48

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	DBL[4:0]					Res	Res	Res	DBA[4:0]				
			RW	RW	RW	RW	RW				RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:13	Reserved			Reserved, must be kept at reset value.
12:8	DBL[4:0]	RW	0 0000	DMA burst length This 5-bit vector defines the length of DMA transfers (the timer recognizes a burst transfer when a read or a write access is done to the TIM16/17_DMAR address), Transfers can be in half-words or in bytes. 00000: 1 transfer 00001: 2 transfers 00010: 3 transfers ... 10001: 18 transfers.
7:5	Reserved	RW	0	Reserved, must be kept at reset value.
4:0	DBA[4:0]	RW	0 0000	DMA base address This 5-bits vector defines the base-address for DMA transfers (when read/write access are done through the TIM16/17_DMAR address). DBA is defined as an offset starting from the address of the TIM16/17_CR1 register. Example: 00000: TIM16/17_CR1 00001: TIM16/17_CR2 00010: TIM16/17_SMCR

20.3.15. DMA address for full transfer (TIM16/17_DMAR)

Address offset: 0x4C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMAB[31:16]															
RW															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DMAB[15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
31:0	DMAB[31:0]	RW	0	DMA register for burst accesses A read or write access to the DMAR register accesses the register located at the Address: "(TIM16/17_CR1 address) + DBA + (DMA index)" in which: TIM16/17_CR1 address is the address of the control register 1, DBA is the DMA base address configured in TIM16/17_DCR register, DMA index is the offset automatically controlled by the DMA transfer, depending on the length of the transfer DBL in the TIM16/17_DCR register.

20.3.16. TIM16/17 register map

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x000	TIMx_CR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CKD[1:0]		ARPE	Res.	Res.	Res.	OPM	URS	UDIS	CEN
	Reset value																							0	0	0				0	0	0	0
0x004	TIM16/17_CR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OIS1N	OIS1	Res.	Res.	Res.	Res.	CCDS	Res.	Res.	CCPC
	Reset value																							0	0					0			0
0x00C	TIM16/17_DIER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC1DE	UDE	BIE	Res.	COMIE	Res.	Res.	Res.	CC1IE	UIE
	Reset value																							0	0	0		0				0	0
0x010	TIM16/17_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC1OF	Res.	BIF	Res.	COMIF	Res.	Res.	Res.	CC1IF	UIF
	Reset value																							0		0		0				0	0
0x014	TIM16/17_EGR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BG	Res.	COMG	Res.	Res.	Res.	CC1G	UG
	Reset value																									0		0				0	0
0x018	TIM16/17_CCMR1 (output compare mode)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC1M [2:0]			OC1PE	OC1FE	CC1S[1:0]	
	Reset value																										0	0	0	0	0	0	0
0x018	TIM16/17_CCMR1 (Input Capture mode)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IC1F [3:0]				IC1PSC[1:0]		CC1S[1:0]	
	Reset value																									0	0	0	0	0	0	0	0
0x020	TIM16/17_CCER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC1NP	CC1NE	CC1P	CC1E
	Reset value																													0	0	0	0

0x24	TIM16/17_CNT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CNT[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x28	TIM16/17_PSC	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PSC[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x2C	TIM16/17_ARR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ARR[15:0]															
	Reset value																	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x30	TIM1_RCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REP[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x34	TIM1_CCR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR1[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x44	TIM1_BDTR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MOE	AOE	BKP	BKE	OSSR	OSSI	LOCK[1:0]		DTG[7:0]							
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x48	TIM1_DCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBL[4:0]					Res.	DBA[4:0]							
	Reset value																			0	0	0	0	0		0	0	0	0	0	0	0	0
0x4C	TIM1_DMAR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DMAB[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

21. Infrared interface (IRTIM)

An infrared interface (IRTIM) for remote control is available on the device. It can be used with an infrared LED to perform remote control functions.

To generate the infrared remote control signals, the IR interface must be enabled and TIM16 channel 1 (TIM16_OC1) and TIM17 channel 1 (TIM17_OC1) must be properly configured to generate correct waveforms. The infrared receiver can be implemented easily through a basic input capture mode.

All standard IR pulse modulation modes can be obtained by programming the two timer output compare channels.

TIM17 is used to generate the high frequency carrier signal, while TIM16 generates the modulation envelope.

The infrared function is output on the IR_OUT pin. The activation of this function is done through the GPIOx_AFRx register by enabling the related alternate function bit.

The high sink LED driver capability (only available on the PB9 pin) can be activated through the I2C_PB9_FMP bit in the SYSCFG_CFGR1 register and used to sink the high current needed to directly control an infrared LED.

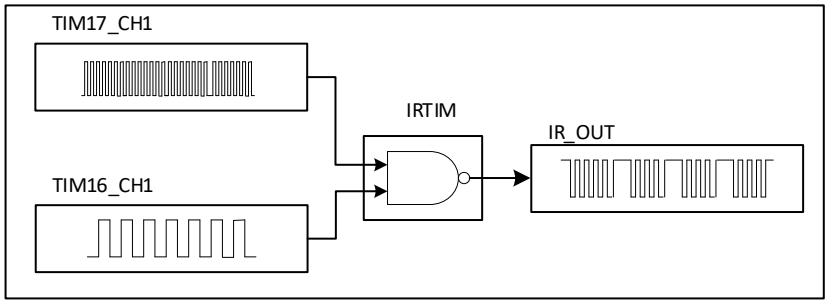

Figure 21-1 Using the infrared function

22. Low power timer (LPTIM)

22.1. Introduction

LPTIM is a 16-bit timer. The ability of LPTIM to wake up the system from a low-power mode makes it suitable for implementing low-power applications.

LPTIM introduces a flexible clocking scheme that provides the required functionality and performance while minimizing power consumption.

22.2. LPTIM main features

- 16-bit up counter
- 3-bit prescaler with 8 possible division factors (1, 2, 4, 8, 16, 32, 64, 128)
- Optional clock
  - Internal clock source: LSE, LSI or APB clock
- 16-bit ARR reload register
- Single mode

22.3. LPTIM functional description

22.3.1. LPTIM block diagram

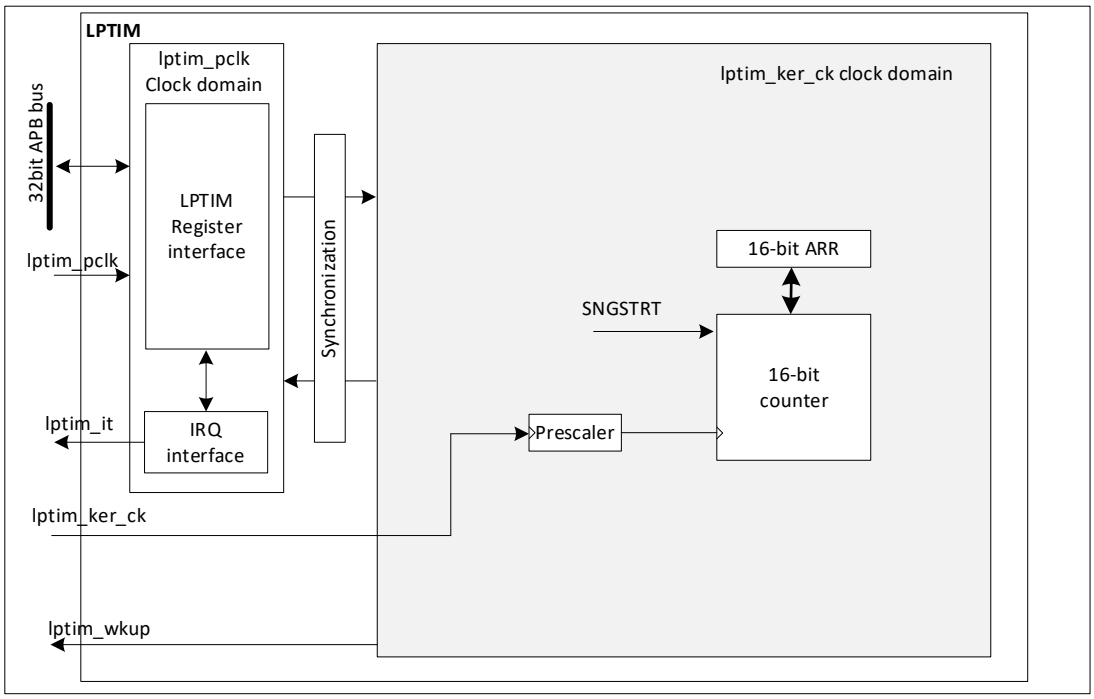

Figure 22-1 LPTIM block diagram

22.3.2. LPTIM pins and internal signals

Table 22-1 LPTIM Internal Signals

Names	Signal type	Description
lptim_pclk	Digital input	LPTIM APB clock domain
lptim_ker_ck	Digital input	LPTIM kernel clock
lptim_it	Digital output	LPTIM global interrupt
lptim_wakeup	Digital output	LPTIM wakeup event

22.3.3. LPTIM reset and clock

LPTIM can use multiple clock sources for count.

Through the RCC module, it can be clocked with an internal clock signal (the clock signal can be selected among APB, LSI, LSE sources).

22.3.4. Prescaler

LPTIM 16-bit counter driven by a configurable 2 power prescaler control. The prescaler division ratio is controlled by PRESC[2:0].

The following table lists all cases:

Table 22-2 prescale factor

Programming	Dividing factor
000	/1
001	/2
010	/4
011	/8
100	/16
101	/32
110	/64
111	/128

22.3.5. Operating mode

LPTIM has only one use timer mode.

- **Single mode**: The timer starts from a trigger event and stops when the ARR value is reached.

To enable single counting, the SNGSTRT bit must be set.

A new trigger event will restart the timer. Any trigger events after the counter has started and before the ARR is reached will be ignored.

22.3.6. Register update

The PRELOAD bit controls how the LPTIM_ARR register is updated:

- When the PRELOAD bit is reset to '0': The LPTIM_ARR register is updated immediately after any write access.
- When the PRELOAD bit is set to '1': If the timer has already started, LPTIM_ARR will be updated at the end of the current period.

The LPTIM APB interface and the LPTIM Kernel logic use different clocks, so there is a certain delay when the APB is written and the written value is applied to the counter comparator. During this delay period, any additional writes to these registers must be avoided.

22.3.7. Enable timer

The ENABLE bit in the LPTIM_CR register is used to enable/disable the LPTIM core logic. After setting the ENABLE bit, a delay of two counter clocks is required to enable LPTIM.

The LPTIM_CFGR and LPTIM_IER registers can only be modified when LPTIM is disabled.

22.3.8. Counter reset INDANG

In order to reset the contents of the LPTIM_CNT register, a reset mechanism is provided:

Asynchronous reset mechanism:

Asynchronous reset is controlled by the RSTARE bit in the LPTIM_CR register. When this bit is set to 1, any access to the LPTIM_CNT register resets its contents to zero.

Note that in order to reliably read the LPTIM_CNT register, two read accesses must be performed and the results are compared. If the results are consistent, the read value is considered to be reliable.

Notice:

- When asynchronous reset is enabled, the first read will reset the LPTIM_CNT, the second read can read the count result of the LPTIM_CNT register.
- When the LPTIM count clock selects PCLK/HSI, the read value cannot be guaranteed to be reliable even if it is accessed twice in a row.

22.3.9. Debug mode

When the microcontroller enters debug mode, the LPTIM counter either continues to work normally or stops, depending on DBG_TIMx_STOP configuration bit in DBG module.

22.4. LPTIM low power mode

Table 22-3 The difference between different low power modes of LPTIM

Mode	Description
Sleep	No effect. LPTIM interrupts cause the device to exit Sleep mode.
Stop	No effect when LPTIM is clocked by LSE or LSI. LPTIM interrupts cause the device to exit Stop.

22.5. LPTIM interrupt

The following events will generate interrupt/wake-up events if they are enabled in the LPTIM_IER register:

- Auto-reload match

Note: If the corresponding bit in the LPTIM_IER register (interrupt enable register) is set to 1 after the corresponding flag in the LPTIM_ISR register (status register) is set to 1, no interrupt will be generated.

Interrupt event	Description
Auto-reload match	When the content of the counter register (LPTIM_CNT) matches the content of the auto-reload register (LPTIM_ARR), the interrupt flag is set

22.6. LPTIM register

22.6.1. LPTIM interrupt and status register (LPTIM_ISR)

Address offset: 0x000

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**

Res	Res	Res	Res	Res	Res	Res	Res	Res						ARRM	
														r	

Bit	Name	R/W	Reset Value	Function
31:2	Reserved	-	0	
1	ARRM	R	0	Auto-reload match ARRM is set by hardware to inform the application that the LPTIM_CNT register value matches the LPTIM_ARR register value. Writing a 1 to the ARRMCF bit of the LPTIM_ICR register clears the ARRM flag.
0	Reserved			

22.6.2. LPTIM interrupt clear register (LPTIM_ICR)

Address offset: 0x004

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	ARRMCF	Res
														w	

Bit	Name	R/W	Reset Value	Function
31:2	Reserved	-	0	
1	ARRMCF	RW	0	Auto-reload match clear flag Writing a 1 to this bit clears the ARRM flag in the LPTIM_ISR register
0	Reserved			

22.6.3. LPTIM interrupt enable register (LPTIM_IER)

Address offset: 0x008

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**~~3~~**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	ARRMIE	Res
														RW	

Bit	Name	R/W	Reset Value	Function
31:2	Reserved	-	0	
1	ARRMIE	RW	0	Auto-reload match interrupt enable 0: ARRM interrupt disabled 1: ARRM interrupt enabled
0	Reserved			

22.6.4. LPTIM configuration register (LPTIM_CFGR)

Address offset:0x00C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	PRE-LOAD	Res	Res	Res	Res	Res	Res

									RW						
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	PRESC[2:0]	Res	Res	Res	Res	Res	Res	Res	Res	Res		
				RW	RW	RW									

Bit	Name	R/W	Reset Value	Function
31:23	Reserved	-	0	
22	PRELOAD	RW	0	Register update mode The preload bit controls the LPTIM_ARR register update mode 0: Update registers after each APB bus write access 1: Registers are updated at the end of the current LPTIM period
21:12	Reserved			
11:9	PRESC[2:0]	RW	0	clock prescaler The PRESC bits configure the prescaler division factor. It can be a factor in the following divisions: 000:/1 001:/2 010:/4 011:/8 100:/16 101:/32 110:/64 111:/128
8:0	Reserved	-	0	Reserved, must be kept at reset value.

22.6.5. LPTIM control register (LPTIM_CR)

Address offset: 0x010

Reset value: 0x0000 0000

31	**30**	**29**	**28**	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	RSTARE	Res	Res	SNGSTRT	ENABLE
											RW			RW	RW

Bit	Name	R/W	Reset Value	Function
31:5	Reserved	-	0	
4:2	RSTARE	RW	0	Reset enable after read This bit is set and cleared by software. When RSTARE is set to '1', any read access to the LPTIM_CNT register will asynchronously reset the LPTIM_CNT register contents.
1	SNGSTRT	RW	0	LPTIM starts single mode. This bit is set by software and cleared by hardware. Setting this bit will start LPTIM in single-pulse mode. Note: This bit can only be set when LPTIM is enabled. It will be reset automatically by hardware.
0	ENABLE	RW	0	LPTIM enable bit, set and cleared by software 0: LPTIM disabled 1: LPTIM enabled

22.6.6. LPTIM auto-reload register (LPTIM_ARR)

Address offset: 0x018

Reset value: 0x0000 0001

31	**30**	**29**	**28**	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**

ARR[15:0]
RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	0	Reserved, must be kept at reset value.
15:0	ARR	RW	0x0001	Auto-reload value ARR is the auto-reload value of LPTIM This register can only be updated when LPTIM is enabled

22.6.7. LPTIM counter (LPTIM_CNT)

Address offset: 0x01C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CNT[15:0]															
R															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	0	
15: 0	CNT	R	0	counter value When LPTIM is running on an asynchronous clock, reading the LPTIM_CNT register may return unreliable values. So in this case it is necessary to perform two consecutive read accesses and verify that the two values returned are the same. A read access can be considered reliable when the values of two consecutive read accesses are equal.

22.6.8. LPTIM register map

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x000	LPTIM_ISR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ARRM	Res.
	Reset value																															0	
0x004	LPTIM_ICR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ARRMC	Res.
	Reset value																															0	
0x008	LPTIM_IER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ARRMI	Res.
	Reset value																															0	
0x00C	LPTIM_CFGR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRELOAD	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRESC [2:0]			Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset										0											0	0	0									

	value																																
0x10	LPTIM_CR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RSTARE	Res.	Res.	SNGSTR	ENABLE
	Reset value																											0			0	0
0x18	LPTIM_ARR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ARR[15:0]																
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0x1C	LPTIM_CNT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CNT[15:0]																
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

23. Independent watchdog (IWDG)

23.1. Introduction

Independent watchdog is integrated in the chip, and this module has the characteristics of high-security level, accurate timing and flexible use. IWDG finds and resolves functional malfunctions due to software failure and triggers system reset when the counter reaches the specified timeout value.

The IWDG is clocked by LSI and thus stay active even if the main clock fails.

The IWDG is the best suited to applications that require the watchdog to run as a totally independent process outside the main application, without having high timing accuracy constraints.

23.2. IWDG main features

- Free-running downcounter
- Clocked from an independent RC oscillator (can operate in Standby and Stop modes)
- Conditional reset
  - Reset when the downcounter value of 0x000 is reached

23.3. IWDG functional description

23.3.1. IWDG block diagram

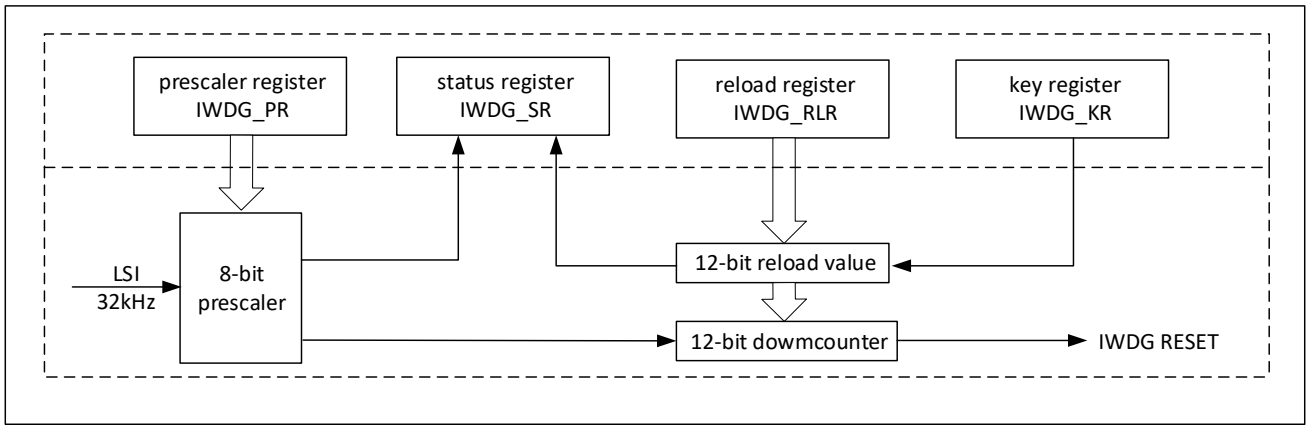

Figure 23-1 IWDG block diagram

When the independent watchdog is started by writing the value 0x0000 CCCC in the Key register (IWDG_KR), the counter starts counting down from the reset value of 0xFFF. When it reaches the end of count value (0x000) a reset signal is generated (IWDG reset).

Whenever the key value 0x0000 AAAA is written in the IWDG_KR register, the IWDG_RLR value is reloaded in the counter and the watchdog reset is prevented.

Once it starts running, IWDG cannot be stopped.

23.3.2. Hardware watchdog

If the “Hardware watchdog” feature is enabled through the device option bits, the watchdog is automatically enabled at power-on, and generates a reset unless the Key register is written by the software before the counter reaches end of count.

23.3.3. Register access protection

Write access to the IWDG_PR and IWDG_RLR registers is protected. A write access to this register with a different value will break the sequence and register access will be protected again. This implies that it is the case of the reload operation (writing 0x0000 AAAA).

A status register is available to indicate that an update of the prescaler or the down-counter reload value is on going.

23.3.4. Debug mode

This function can only be used if the system supports DBG_MCU.

When the microcontroller enters debug mode, the IWDG counter either continues to work normally or stops, depending on DBG_IWDG_STOP configuration bit in DBG module.

23.4. IWDG registers

23.4.1. Key register (IWDG_KR)

Address offset: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
KEY[15:0]															
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	RES	-	Reserved
15:0	KEY[15:0]	W	0x00	Key value. These bits must be written by software at regular intervals with the key value 0XAAAA, otheRWise the watchdog generates a reset when the counter reaches 0. Writing the key value 0x5555 to enable access to the IWDG_PR and IWDG_RLR registers. Writing the key value 0xCCCC starts the watchdog (except if the hardware watchdog option is selected)

23.4.2. Prescaler register (IWDG_PR)

Address offset: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	PR[2:0]		
													RW		

Bit	Name	R/W	Reset Value	Function
31:3	Reserved	RES	-	Reserved

2:0	PR[2:0]	RW	0	Prescaler divider. They are select the prescaler divider feeding the counter clock by set this register. PVU bit of IWDG_SR must be reset in order to be able to change the prescaler divider. 000: divider /4 001: divider /8 010: divider /16 011: divider /32 100: divider /64 101: divider /128 110: divider /256 111: divider /256

23.4.3. Reload register (IWDG_RLR)

Address offset: 0x08

Reset value: 0x0000 0FFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	RL[11:0]											

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	RES	-	Reserved
11:0	RL[11:0]	RW	0	IWDG counter reload value. RL value will be loaded in the counter each time when the value 0xAAAA is writtern in the IWDG_KR register. The watchdog counter counts down from this value. The timeout period is a function of this value and the clock prescaler. The RVU bit in the IWDG_SR register must be reset in order to be able to change the reload value.

23.4.4. Status register (IWDG_SR)

Address offset: 0x0C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	RVU	PVU
														R	R

Bit	Name	R/W	Reset Value	Function
31:2	Reserved	RES	-	Reserved
1	RVU	R	0	Watchdog counter reload value update This bit is set by hardware to indicate that an update of the reload value is ongoing. It is reset by hardware when the reload value update operation is completed.
0	PVU	R	0	Watchdog prescaler value update This bit is set by hardware to indicate that an update of the prescaler value is ongoing. It is reset by hardware when the prescaler update operation is completed.

Note: It is mandatory to wait until RVU bit is reset before changing the IWDG_RLR, to wait until PVU bit is reset before changing the IWDG_PR. However, after updating the IWDG_PR and/or the IWDG_RLR, it is not necessary to wait until IWDG_SR.PVU or IWDG_SR.RVU is reset before continuing code execution

23.4.5. IWDG register map

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	IWDG_KR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	KEY[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x04	IWDG_PR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PR[2:0]		
																															0	0	0
0x08	IWDG_RLR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RL[11:0]											
																						0	0	0	0	0	0	0	0	0	0	0	0
0x0C	IWDG_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RVU	PVU
																															0	0	0

24. System window watchdog (WWDG)

24.1. Introduction

The system window watchdog (WWDG) is used to detect the occurrence of a software fault, usually generated by external interference or by unforeseen logical conditions, which causes the application program to abandon its normal sequence. The watchdog circuit generates an MCU reset on expiry of a programmed time period, unless the program refreshes the contents of the downcounter before the T6 bit becomes cleared.

An MCU reset is also generated if the 7-bit downcounter value (in the control register) is refreshed before the downcounter has reached the window register value. This implies that the counter must be refreshed in a limited window.

The WWDG clock is prescaled from the APB clock and has a configurable time-window that can be programmed to detect abnormally late or early application behavior.

The WWDG is best suited for applications which require the watchdog to react within an accurate timing window.

24.2. WWDG main features

- Programmable free-running downcounte
- Conditional reset
  - Reset when the downcounter value becomes less than 0x40
  - Reset if the downcounter is reloaded outside the window
- Early wakeup interrupt(EWI): triggered (if enabled and the watchdog activated) when the downcounter is equal to 0x40.

24.3. WWDG functional description

If the watchdog is activated (the WDGA bit is set in the WWDG_CR register) and when the 7-bit downcounter (T[6:0] bits) is decremented from 0x40 to 0x3F (T6 becomes cleared), it initiates a reset. If the software reloads the counter while the counter is greater than the value stored in the window register, then a reset is generated.

The application program must write in the WWDG_CR register at regular intervals during normal operation to prevent an MCU reset. This operation must occur only when the counter value is lower than the window register value and higher than 0x3F. The value to be stored in the WWDG_CR register must be between 0xFF and 0xC0.

24.3.1. WWDG block diagram

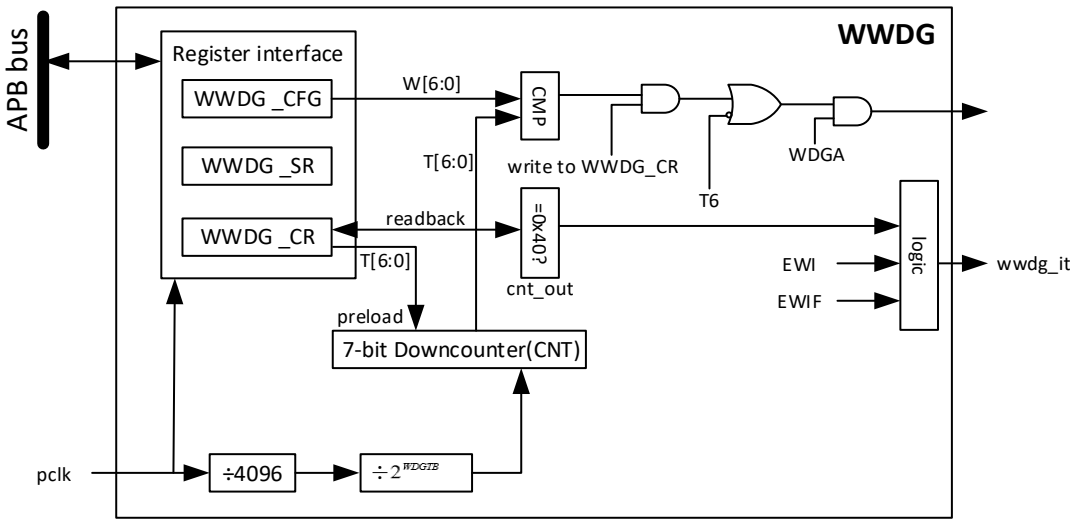

Figure 24-1 Window watchdog block diagram

24.3.2. Enabling the watchdog

When the user selects "software WWDG" in the WWDG_SW bit in the option byte, the watchdog is usually disabled after reset. Then by setting the WDGA bit in the WWDG_CR register, the WWDG module is enabled and then cannot be disabled unless a reset occurs.

When the user selects "hardware WWDG" in WWDG_SW in the option byte, the module is usually enabled after reset and cannot be disabled.

24.3.3. Controlling the downcounter

This downcounter is free-running, counting down even if the watchdog is disabled. When the watchdog is enabled, the T6 bit must be set to prevent generating an immediate reset.

The T[5:0] bits contain the number of increments which represents the time delay before the watchdog produces a reset.

The configuration register (WWDG_CFR) contains the high limit of the window: to prevent a reset, the downcounter must be reloaded when its value is lower than the window register value and greater than 0x3F.

Another way to reload the counter is to use the Early Wakeup Interrupt (EWI). This interrupt is enabled by setting the EWI bit in the WWDG_CFR register. When the down counter reaches 0x40, this interrupt is generated, and the corresponding interrupt service routine (ISR) can be used to load the counter to prevent the WWDG from being reset. This interrupt can be cleared by writing '0' in the WWDG_SR register.

24.3.4. Advanced watchdog interrupt feature

The Early Wakeup Interrupt (EWI) can be used if specific safety operations or data logging must be performed before the actual reset is generated. The EWI interrupt is enabled by setting the EWI bit in the WWDG_CFR register. When the downcounter reaches the value 0x40, an EWI interrupt is generated and the corresponding interrupt service routine (ISR) can be used to trigger specific actions (such as communications or data logging), before resetting the device.

24.3.5. How to program the watchdog timeout

When writing to the WWDG_CR register, always write 1 in the T6 bit to avoid generating an immediate reset

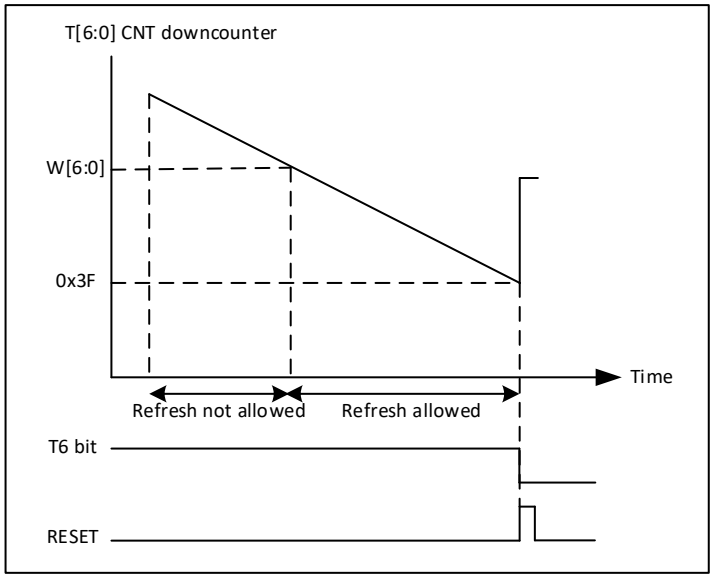

Figure 24-2 Window watchdog timing diagram

The formula to calculate the timeout value is given by:

$$t_{WWDG} = t_{PCLK} \times 4096 \times 2^{WWDGTB[1:0]} \times (T[5:0] + 1) \quad (ms)$$

24.4. WWDG registers

24.4.1. Control register (WWDG_CR)

Address offset: 0x00

Reset value: 0x0000 007F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	WDGA	T[6:0]						
								RS	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	RES	-	Reserved
7	WDGA	RS	0	Activation bit This bit is set by software and only cleared by hardware after a reset. When WDGA = 1, the watchdog can generate a reset. 0: Watchdog disabled 1: Watchdog enabled
6:0	T[6:0]	RW	32'h7F	7-bit counter (MSB to LSB) These bits contain the value of the watchdog counter. It is decremented every (4096x2^{WDGTB}) PCLK cycles. A reset is produced when it is decremented from 0x40 to 0x3F (T6 becomes cleared).

24.4.2. Configuration register (WWDG_CFR)

Address offset: 0x04

Reset value: 0x0000 007F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	EWI	WDGTB[1:0]		T[6:0]						
						RS	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	RES	-	Reserved
9	EWI	RS	0	Early wakeup interrupt When set, an interrupt occurs whenever the counter reaches the value 0x40. This interrupt is only cleared by hardware after a reset.
8:7	WDGTB[1:0]	RW	2'b0	Timer base The time base of the prescaler can be modified as follows: 00: CK Counter Clock (PCLK div 4096) div 1 01: CK Counter Clock (PCLK div 4096) div 2 10: CK Counter Clock (PCLK div 4096) div 4 11: CK Counter Clock (PCLK div 4096) div 8
6:0	W[6:0]	RW	7'h7F	7-bit window value These bits contain the window value to be compared to the downcounter.

24.4.3. Status register (WWDG_SR)

Address offset: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	EWIF
															RC_W0

Bit	Name	R/W	Reset Value	Function
31:1	Reserved	RES	-	Reserved
0	EWIF	RC_W0	0	Early wakeup interrupt flag This bit is set by hardware when the counter has reached the value 40h. It must be cleared by software by writing '0'. A write of '1' has no effect. This bit is also set if the interrupt is not enabled.

24.4.4. WWDG register map

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x000	WWDG_CR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WDGA	T[6:0]						
	Reset value																									0	1	1	1	1	1	1	1
0x004	WWDG_CFR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EWI	WDGTB[1:0]	T[6:0]							
	Reset value																							0	0	0	1	1	1	1	1	1	1
0x	WWDG	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EWIF

08	_SR																													
	Reset value																													0

25. Real-time clock (RTC)

25.1. Introduction

The real-time clock (RTC) is an independent timer/counter. The RTC module has a set of continuous counting counters, which can provide the function of clock-calendar under the corresponding software configuration. Modifying the value of the counter can reset the current time and date of the system.

25.2. RTC main features

- Programmable prescale factor: up to 2^{20}
- 32-bit programmable counter for longer time period measurement
- Two separate clocks: PCLK and RTC clock for APB interface (PCLK clock frequency is more than four times faster than RTC clock frequency)
- Three RTC clock sources:
  - HSE clock divided by 128
  - LSE oscillator clock
  - LSI oscillator clock
- Three dedicated maskable interrupts:
  - Alarm interrupt, used to generate a software programmable alarm interrupt
  - Second interrupt, Used to generate a programmable periodic interrupt signal (up to 1 second)
  - Overflow interrupt, indicating that the internal programmable counter overflows and rolls back to 0

25.3. RTC functional description

25.3.1. Overview

The RTC consists of two main parts (see diagram below). The first part (APB interface) is used to connect with the APB bus. The other part (RTC core) consists of a set of programmable counters, divided into two main modules:

The first module is the prescaler module of the RTC, which can be programmed to generate the RTC time base TR_CLK up to 1 second. The RTC prescaler module contains a 20-bit programmable divider (RTC prescaler). The RTC generates an interrupt (seconds interrupt) in every TR_CLK cycle if the corresponding enable bit is set in the RTC_CR register.

The second module is a 32-bit programmable counter that can be initialized to the current system time. The system time is accumulated in TR_CLK cycles and compared with the programmable time stored in the RTC_ALR register. If the corresponding enable bit is set in the RTC_CR control register, an alarm interrupt will be generated on a compare match.

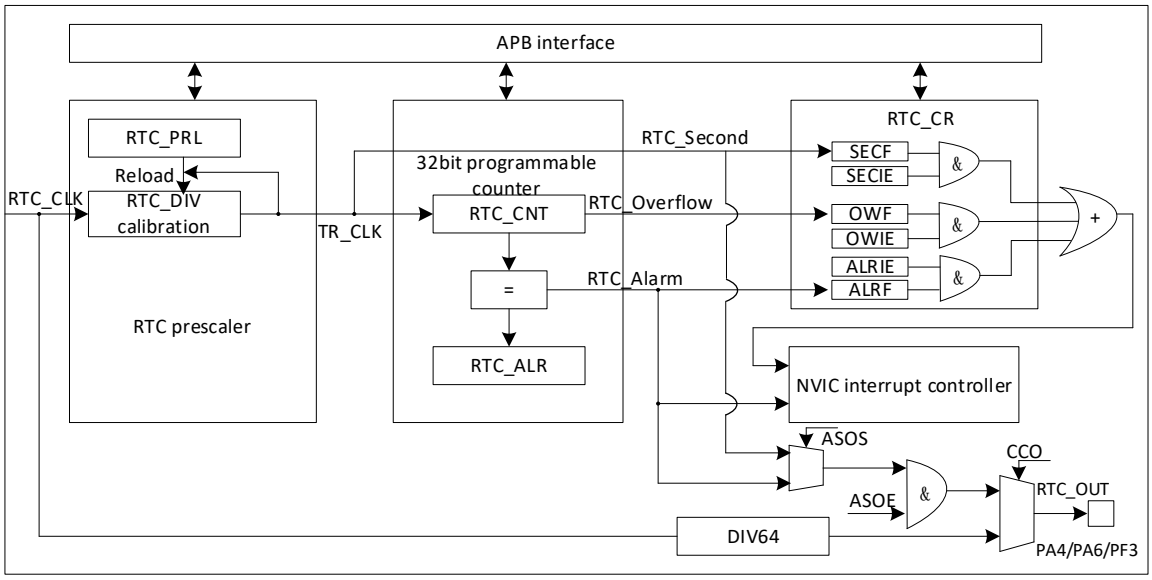

Figure 25-1 RTC block diagram

The DBP bit of the PWR_CR1 register is used to control the write protection disable to the RTC register. By default, DBP = 0, the RTC register cannot be written and accessed, and the RTC register can be written only after the software sets the DBP.

25.3.2. Resetting RTC register

All registers of RTC are reset from power-on reset (POR/PDR/BOR) and software reset of RTC (RCC_BDCR.16), and the rest of reset sources (NRST/IWDG/WWDG/OBL/SYSRESETREQ) have no effect.

Note that when the reset source for the RTC module cannot be generated, not only the RTC module cannot be reset, but also the clock source enable signal sent to the RTC module cannot be reset, nor the clock source selection control sent to the RTC module.

25.3.3. Reading RTC register

The RTC core and the APB clock are completely independent. The values of the RTC_DIV, RTC_CNT, and RTC_ALR registers accessed by the configuration software are all through the APB interface, but the update of the relevant readable registers is internally first passed through the rising edge of the RTC clock, and then synchronized again using the APB clock. The same is true for the flag register of the RTC.

If the APB interface has been disabled before, and the read operation is performed immediately after the APB interface is enabled (at this time, before the register is updated for the first time), the read operation may be corrupted (usually read returns 0). This situation will occur in the following situations:

- System reset or power-on reset
- The chip just exits stop mode

In all the above-mentioned situations, the RTC core circuit keeps running, and the APB interface is turned off (reset or no clock).

Correspondingly, when reading the RTC register, after the RTC APB interface has been closed, the software must wait for the RSF bit (register synchronization flag) of the RTC_CRL register to be set by hardware.

25.3.4. Configuring RTC register

The RTC_PRL, RTC_CNT, RTC_ALR registers can only be written to by entering the configuration mode by setting the CNF bit of the RTC_CRL register.

In addition, writing to any RTC register is only enabled when the previous write operation is finished. To enable the software to detect this condition, the chip provides RTOFF status bit in the RTC_CR register, which is used to display the update status of the register. A new value can be written to the RTC register only when the RTOFF status bit is 1.

Configuration process

1. Poll RTOFF until the bit goes high
2. Set CNF bit to enter configuration mode
3. Write 1 or more RTC registers
4. Clear CNF bit to exit configuration mode
5. Poll RTOFF, wait until it changes to 1 (check for end of write operation).

The write operation is performed only when the CNF bit is cleared, and at least 3 RTC_CLK cycles are required to complete the write operation.

25.3.5. RTC flag assertion

On every RTC clock cycle, before changing the RTC counter, the hardware sets the RTC seconds flag (SECF). On the last RTC clock cycle before the counter reaches 0x0000, the RTC overflow flag (OWF) is set.

RTC_Alarm and RTC alarm flag (ALRF) are set in the RTC clock cycle before the value of the counter reaches the value of the alarm register plus 1 (RTC_ALR+1). Writes to the RTC alarm must be synchronized with the RTC seconds flag using one of the following procedures:

(1) Use the RTC alarm interrupt and modify the RTC alarm and/or RTC counter in the interrupt handler.

(2) Wait for the SECF bit in the RTC control register to be set before changing the RTC alarm and/or the RTC counter.

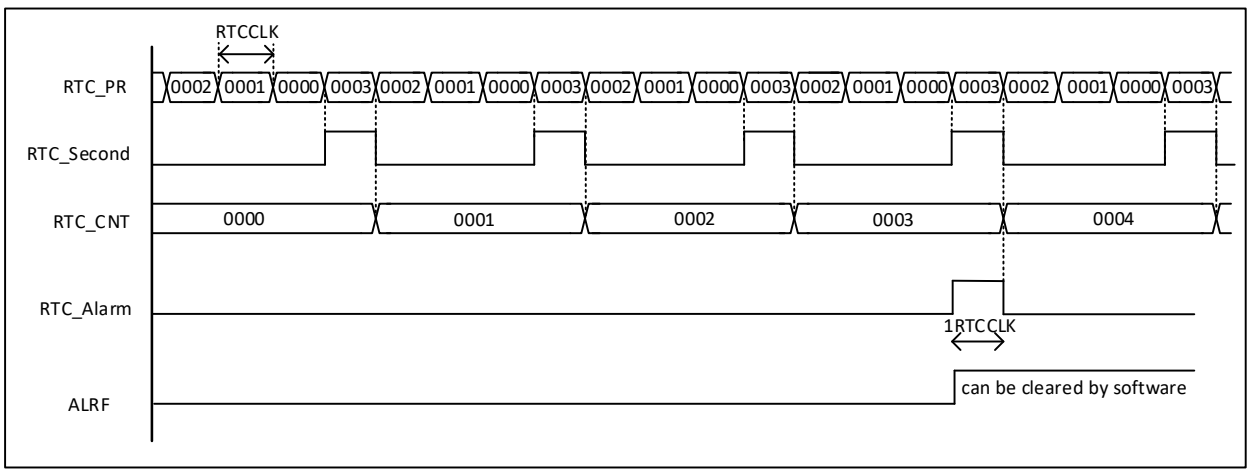

Figure 25-2 Example of RTC seconds and alarm waveforms, PR = 0003, ALARM = 00004

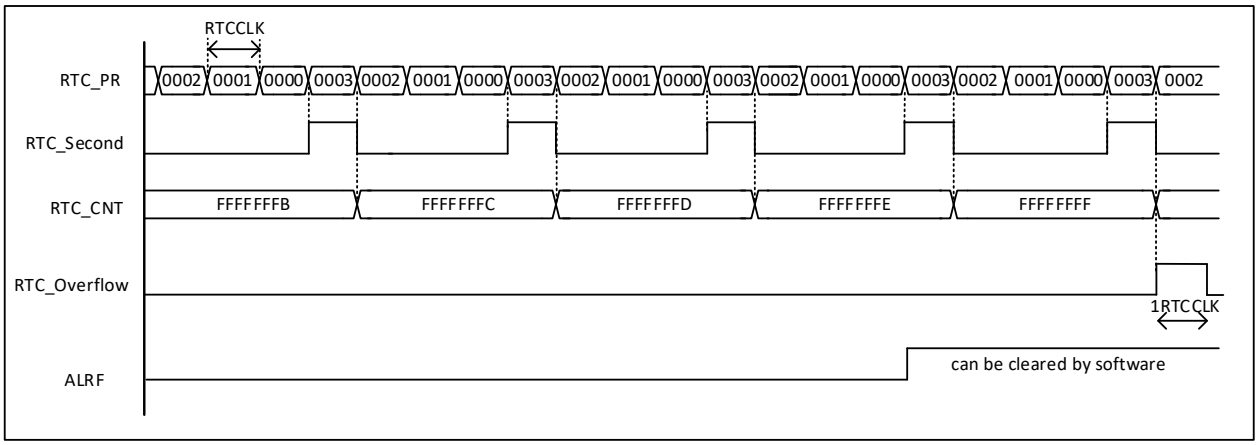

Figure 25-3 RTC overflow waveform example, PR = 0003

25.3.6. RTC calibration

For measurement purposes, the RTC clock divided by 64 can be output on the IO pin (PA4). This function is achieved by setting the CCO bit (RTCCR register).

By configuring the CAL[6:0] bits, the clock can be slowed down to 121 PPM.

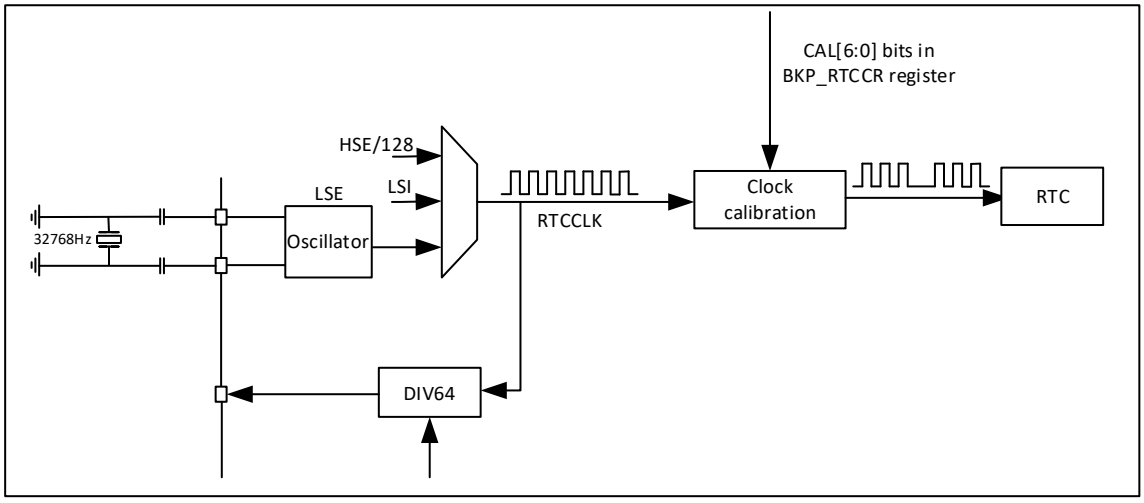

Figure 25-4 RTC calibration chart

25.4. RTC registers

25.4.1. RTC control register high bit (RTC_CRH)

Address offset: 0x00

Reset value: 0x0000

When PWR_CR1.DBP is 1, it is allowed to write to this register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	OWIE	ALR IE	SEC IE

													RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:3	Reserved			
2	OWIE	RW	0	Overflow interrupt enable bit 0: Overflow interrupt disable 1: Overflow interrupt enable
1	ALRIE	RW	0	Alarm interrupt enable bit 0: Alarm interrupt disabled 1: Alarm interrupt enabled
0	SECIE	RW	0	Second interrupt enable bit 0: Second interrupt disable 1: Second interrupt enable

These bits are used to mask interrupt requests. Note that, all interrupts are masked after system reset, so it is possible to ensure that there are no pending interrupt requests after initialization by writing to the RTC register. When the peripheral is completing the previous write operation (the flag bit RTOFF = 0), the RTC_CRH register cannot be written.

The RTC function is controlled by this control register. Some certain bits must use a dedicated configuration flow to be written

25.4.2. RTC control registe low bit (RTC_CRL)

Address offset: 0x04

Reset value: 0x0020

When PWR_CR1.DBP is 1, it is allowed to write to this register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	RTOFF	CNF	RSF	OWF	ALRF	SECF
										R	RW	RC_W0	RC_W0	RC_W0	RC_W0

Bit	Name	R/W	Reset Value	Function
31:6	Reserved			
5	RTOFF	R	1	RTC operation OFF, this bit is read only. This bit is used by the RTC module to indicate the status of the last operation on its registers (indicating whether the operation is complete). If this bit is '0', it means that no RTC register can be written. 0: The last write operation to the RTC register is still in progress 1: The last write operation to the RTC register has been completed
4	CNF	RW	0	Configuration flag This bit must be set to '1' by software to enter configuration mode, allowing new values to be written to the RTC_CNT, RTC_ALR or RTC_PRL registers. Only after this bit is set to '1' and cleared to '0' again by software, the write operation will be performed. 0: Exit configuration mode (start to update RTC register) 1: Enter configuration mode
3	RSF	RC_W0	0	Registers synchronized flag When the RTC_CNT register and the RTC_DIV register are updated, the hardware sets this bit to '1', and the software clears this bit. This bit must be cleared to '0' by software after an APB reset, or after the APB clock is stopped. Before performing any read operations, the user program must wait for this bit to be set to '1' by hardware to ensure

				that RTC_CNT, RTC_ALR or RTC_PRL have been synchronized. 0: Register has not been synchronized 1: Register has been synchronized
2	OWF	RC_W0	0	Overflow flag This bit is set to '1' by hardware when the 32-bit programmable counter overflows. If OWIE = 1 in the RTC_CRH register, an interrupt will be generated. This bit can only be cleared to '0' by software, writing '1' has no effect. 0: No overflow 1: 32-bit programmable counter overflow
1	ALRF	RC_W0	0	Alarm flag When the 32-bit programmable counter reaches the predetermined value set by the RTC_ALR register, this bit is set to '1' by hardware. An interrupt is generated if ALRIE = 1 in the RTC_CRH register. This bit can only be cleared to '0' by software, writing '1' has no effect. 0: No alarm clock 1: There is an alarm clock
0	SECF	RC_W0	0	Second flag When the 32-bit programmable prescaler overflows, this bit is set to '1' by hardware and the RTC counter is incremented by 1. Therefore, this flag provides a periodic signal (usually 1 second) to the resolution programmable RTC counter. An interrupt is generated if SECIE = 1 in the RTC_CRH register. This bit can only be cleared by software, writing '1' has no effect. 0: The second mark condition is not established 1: The second mark condition is established

The function of the RTC is controlled by this control register. When the peripheral is continuing the last write operation (RTOFF = 0), the RTC_CR register cannot be written.

25.4.3. RTC prescaler reload value high (RTC_PRLH)

The PRL register holds the periodic count value of the RTC prescaler. This register is write-protected by RTOFF bit of RTC_CR register, only RTOFF = 1 can write operation.

Address offset: 0x08

Write only

Reset value: 0x0000

Clock source: LSI/LSE/(HSE/128)

When PWR_CR1.DBP is 1, it is allowed to write to this register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	PRL[19:16]			
												W	W	W	W

Bit	Name	R/W	Reset Value	Function
31:4	Reserved			
3:0	PRL[19:16]	W	0	RTC prescaler reload value high These bits are used to define the clock frequency of the counter according to the following formula: fTR_CLK = fRTCCLK/(PRL[19:0]+1) Note: 0 value is not recommended, OtheRWise, the RTC interrupt and flag bit cannot be generated correctly

25.4.4. RTC prescaler reload value low (RTC_PRLL)

Address offset: 0x0C

Write only

Reset value: 0x8000

Clock source: LSI/LSE/HSE

When PWR_CR1.DBP is 1, it is allowed to write to this register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PRL[15:0]															
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			
15:0	PRL	W	0x8000	RTC prescaler reload value high These bits are used to define the clock frequency of the counter according to the following formula: fTR_CLK = fRTCCLK/(PRL[19:0]+1) Note: 0 value is not recommended, OtheRWise, the RTC interrupt and flag bit cannot be generated correctly

25.4.5. RTC prescaler divide register high (RTC_DIVH)

At each TR_CLK cycle, the value of the RTC_PRL register is reloaded into the RTC prescaler counter. In order to get an accurate clock, it is possible to read the current value of the prescaler counter (without stopping it), which is stored in the RTC_DIV register.

This register is read-only, when the value of the RTC_PRL or RTC_CNT register changes, it will be reloaded by (RTC_PRL).

Address offset: 0x10

Reset value: 0x0000

When PWR_CR1.DBP is 1, it is allowed to write to this register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	RTC_DIV[19:16]			
												R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:4	Reverved			
3:0	RTC_DIV[19:16]	R	0	RTC clock dividers

25.4.6. RTC prescaler divide factor register low (RTC_DIVL)

Address offset: 0x14

Reset value: 0x8000

When PWR_CR1.DBP is 1, it is allowed to write to this register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DIV[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:16	Reverved			

15:0	DIV[15:0]	R	0x8000	RTC clock dividers

25.4.7. RTC count register high (RTC_CNTH)

The RTC module has a 32-bit programmable counter, which is accessed through two 16-bit registers and counts based on the TR_CLK generated by the prescaler.

The RTC_CNT register holds the count value of this counter. The registers are write-protected and can only be written when RTOFF = 1. Write to the high 16bit RTC_CNTH or low 16bit RTC_CNTL register, directly load it into the corresponding programmable counter, and reload the RTC prescaler. When a read operation occurs, the current value of the counter (system date) is returned.

Address offset: 0x18

Reset value: 0x0000

When PWR_CR1.DBP is 1, it is allowed to write to this register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RTC_CNT[31:16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			
15:0	RTC_CNT[31:16]	RW	0x0000	The high 16bits of the RTC core counter When reading the RTC_CNTH register, it returns the high 16bits of the current value of the RTC counter register. This register can only be written to by entering configuration mode.

25.4.8. RTC count register low (RTC_CNTL)

Address offset: 0x1C

Reset value: 0x0000

When PWR_CR1.DBP is 1, it is allowed to write to this register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RTC_CNT[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			
15:0	RTC_CNT[15:0]	RW	0x0000	The lower 16 bits of the RTC core counter When reading the RTC_CNTL register, it returns the lower 16 bits of the current value of the RTC counter register. This register can only be written to by entering configuration mode.

25.4.9. RTC alarm register high (RTC_ALRH)

When the programmable counter (count) reaches the 32bit value stored in the RTC_ALR register, an alarm interrupt request is generated. This register is write-protected by the RTOFF bit, and write access is allowed only when RTOFF = 1.

Address offset: 0x20

Reset value: 0xFFFF

When PWR_CR1.DBP is 1, it is allowed to write to this register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RTC_ALR[31:16]															
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			
15:0	ALR[31:16]	W	0xFFFF	RTC alarm high 16bit Software can write the high 16bit of Alarm time. Writing to this register must enter configuration mode.

25.4.10. RTC alarm register low (RTC_ALRL)

Address offset: 0x24

Reset value: 0xFFFF

When PWR_CR1.DBP is 1, it is allowed to write to this register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RTC_ALR[15:0]															
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			
15:0	ALR[15:0]	W	0xFFFF	RTC alarm low 16bit Software can write the low 16bit of Alarm time. Writing to this register must enter configuration mode.

25.4.11. RTC clock calibration register (BKP_RTCCR)

Address offset: 0x2C

Reset value: 0x0000

When PWR_CR1.DBP is 1, it is allowed to write to this register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	ASOS	ASOE	CCO	CAL[6:0]						
						RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved			
9	ASOS	RW	0	Alarm or second output selection When the ASOE bit is set, the ASOS bit can be used to select whether the output on the Pin is RTC second pulse or Alarm pulse signal 0: RTC Alarm pulse signal 1: RTC second pulse signal
8	ASOE	RW	0	Alarm or second output enable When this bit is set, the ASOS bit determines whether the output on the pin is the RTC second pulse or the Alarm pulse signal.
7	CCO	RW	0	Calibration clock output 0: No effect

				1: When this bit is set, the 64 frequency division of the RTC clock is output on the pin
6:0	CAL[6:0]	RW	0	Calibration value This value shows the negligible number of clock pulses per 2^{20} clock pulses, which allows the RTC to calibrate to slow down the clock in steps of $1000000/2^{20}$ PPM. RTC clock can be slowed down from 0 to 121PPM

25.4.12. RTC register map

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x000	RTC_CRH	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OWIE	ALRIE	SECIE
	Reset value																														0	0	0
0x004	RTC_CRL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RTOFF	CNF	RSF	OWF	ALRF	SECF
	Reset value																											1	0	0	0	0	0
0x008	RTC_PRLH	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRL[19:16]			
	Reset value																													0	0	0	0
0x00C	RTC_PRLL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRL[15:0]															
	Reset value																	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x010	RTC_DIVH	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RTC_DIV[19:16]			
	Reset value																													0	0	0	0
0x014	RTC_DIVL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DIV[15:0]															
	Reset value																	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x018	RTC_CNTH	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RTC_CNT[31:16]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x01C	RTC_CNTL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RTC_CNT[15:0]															
	Reset																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	value																														
0x20	RTC_ALRH	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RTC_ALR[31:16]													
	Reset value																	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x24	RTC_ALRL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RTC_ALR[15:0]													
	Reset value																	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x2C	BKP_RTCCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ASOS	ASOE	CCO	CAL[6:0]				
	Reset value																							0	0	0	0	0	0	0	0	0	0

26. Inter-integrated circuit (I2C) interface

26.1. Introduction

The I2C (inter-integrated circuit) bus interface handles communications between the microcontroller and the serial I2C bus. It provides multimaster capability, and controls all I2C bus-specific sequencing, protocol, arbitration and timing. It supports Standard-mode (Sm), Fast-mode (Fm) and Fast-mode Plus (Fm+).
DMA can be used to reduce CPU overload.

26.2. I2C main features

- Slave and master modes
- Multimaster capability: Can be master or slave
- Support different communication speeds
  - Standard-mode: up to 100 kHz
  - Fast-mode: up to 400 kHz
- As Master
  - issue clock
  - issue Start & Stop clock
- As slave
  - Programmable I2C address detection
  - Stop bit discovery
- 7-bit addressing mode
- General call
- Status flag bit
  - Transmit/receive mode flag bit
  - Byte transfer completion flag bit
  - I2C busy flag bit
- error flag bit
  - Host Arbitration Lost
  - ACK failure after address/data transfer
  - Start/Stop error
  - Overrun/Underrun(Clock stretch function disabled)
- Optional clock stretching
- Single-byte buffer with DMA capability
- Software reset
- Analog noise filter function
- Configurable PEC (packet error checking) generation and verification
  - PEC value can be sent in the last bytes in Tx mode
  - The last byte does PEC error checking

26.3. I2C functional description

26.3.1. I2C block diagram

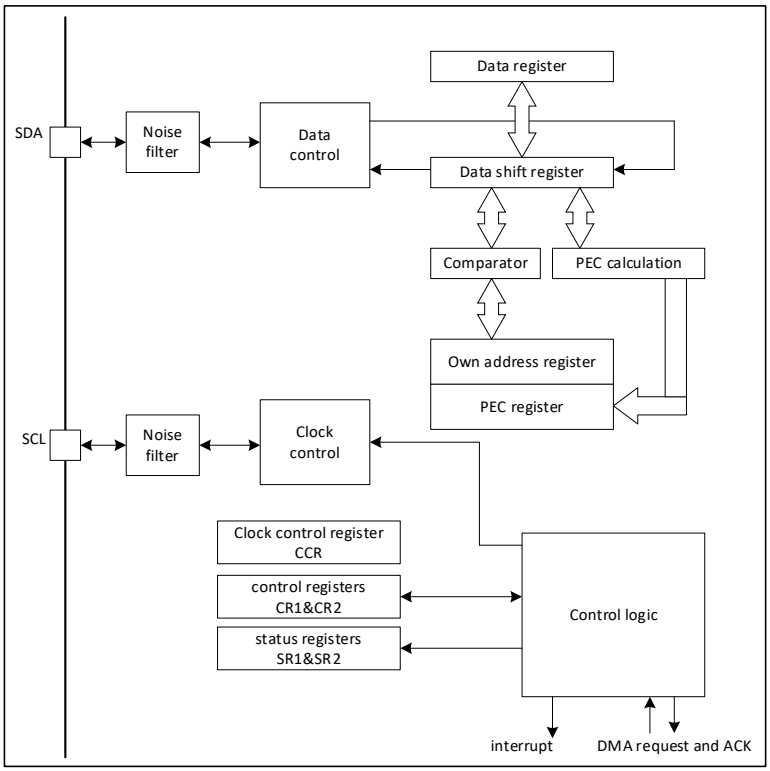

Figure 26-1 I2C block diagram

26.3.2. Mode selection

The interface can operate in one of the four following modes:

- Slave transmitter
- Slave receiver
- Master transmitter
- Master receiver

By default, it operates in slave mode. The interface automatically switches from slave to master when it generates a START condition, and from master to slave if an arbitration loss or a STOP generation occurs, allowing multimaster capability.

26.3.2.1. Communication flow

In Master mode, the I2C interface initiates a data transfer and generates the clock signal. A serial data transfer always begins with a START condition and ends with a STOP condition. Both START and STOP conditions are generated in master mode by software.

In Slave mode, the interface is capable of recognizing its own addresses (7 or 10-bit), and the General Call address. The General Call address detection can be enabled or disabled by software. The reserved SMBus addresses can also be enabled by software.

Data and addresses are transferred as 8-bit bytes, MSB first. The first byte(s) following the START condition contain the address (one in 7-bit mode, two in 10-bit mode). The address is always transmitted in Master mode.

A 9th clock pulse follows the 8 clock cycles of a byte transfer, during which the receiver must send an acknowledge bit to the transmitter. Refer to the following figure.

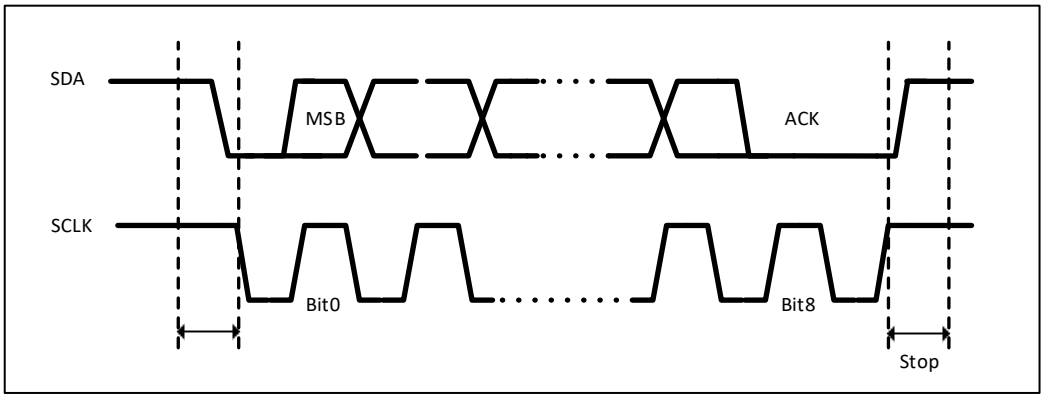

Figure 26-2 I2C bus protocol

Acknowledge can be enabled or disabled by software. The I2C interface addresses can be selected by software.

26.3.3. I2C initialization

26.3.3.1. Enabling and disabling the peripheral

The I2C peripheral clock must be configured and enabled the bit of I2C_EN in the RCC_APBENR1 register. Then the I2C can be enabled by setting the PE bit in the I2C_CR1 register.

26.3.3.2. I2C timings

The timings must be configured in order to guarantee a correct data hold and setup time, used in master and slave modes. This is done by programming the I2C_CCR and I2C_TRISE register.

26.3.4. I2C slave mode

By default, the I2C interface always works in slave mode. To switch from slave mode to master mode, a start condition needs to be generated.

In order to generate the correct timing, the input clock to this module must be programmed in the I2C_CR2 register. The frequency of the input clock must be at least:

Standard mode: 2 MHz

Fast mode: 4 MHz

If start condition is detected,the address received on the SDA line is sent to the shift register and is combined with the chip's address OAR1 or general The call address (if ENGC = 1) is compared.

Header or address mismatch:

The I2C interface ignores it and waits for another start condition.

Address match:

The I2C interface generates the following timings:

● If ACK is set to '1' by software, an acknowledge pulse is generated.

● The ADDR bit is set by hardware, and if the ITEVTEN bit is set, an interrupt is generated.

In slave mode, the TRA bit indicates that it is currently in receiver mode or transmitter mode.

26.3.4.1. Slave transmitter

After receiving the address and clearing the ADDR bit, (if the least significant bit of the address byte is 1) the Slave sends the data (byte) from the DR register to the SDA by the internal shift register.

Slave pulls SCL low until the ADDR bit is cleared and the data to be transmitted has been written to the DR register.

When an acknowledge pulse is received: The TxE bit is set by hardware and an interrupt is generated if the ITEVTEN and ITBUFEN bits are set.

If the TxE bit is set, but no new data is written to the I2C_DR register before the end of the next data transmission, the BTF bit is set. Slave pulls SCL low until the BTF bit is cleared by software (after reading _SR1, then writing to the I2C_DR register).

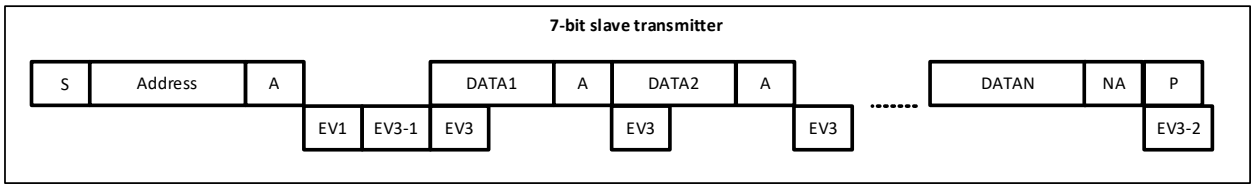

Figure 26-3 Transmission sequence diagram from the sender

Legend: S = Start, Sr = Repeated Start, P = Stop, A = Acknowledge, NA = Non-acknowledge, EVx = Event (interrupt when ITEVFEN = 1)

EV1: ADDR = 1, by first reading the SR1 register, then reading the SR2 register to clear the ADDR bit

EV3-1: TxE = 1, shift register empty, data register empty, write Data1 to DR register

EV3: TxE = 1, shift register is not empty, data register is empty, write to DR register (Data2) to clear TxE

EV3-2: AF = 1, software write 0 to AF bit to clear this bit

26.3.4.2. Slave receiver

After receiving the address and clearing ADDR, (if the least significant bit of the address byte is 0) the slave will store the byte received from the SDA line into the DR register by the internal shift register. The I2C interface performs the following actions after each byte is received:

● If the ACK bit is set, an acknowledge pulse is generated

● The hardware sets RxNE = 1. An interrupt is generated if the ITEVTEN and ITBUFEN bits are set.

If RxNE is set and the DR register is not read before the end of receiving new data, the BTF bit is set, and the slave keeps pulling SCL low until the BTF is cleared (the I2C_DR register is read after I2C_SR1). (See below).

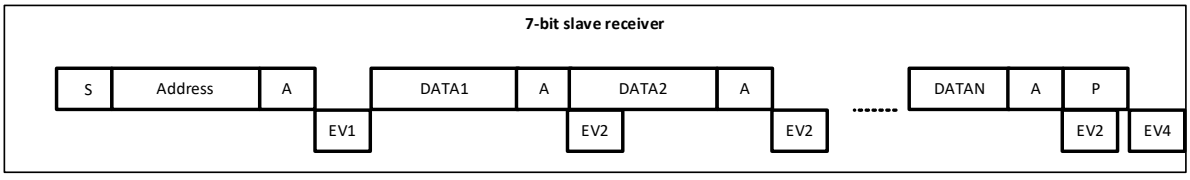

Figure 26-4 Slave receiver transmission sequence diagram

Legend: S = Start, Sr = Repeated Start, P = Stop, A = Acknowledged, EVx = Event(with interrupt if ITEVFEN = 1)

EV1: ADDR = 1, by reading SR1 first, then SR2, the ADDR is cleared

EV2: RxNE = 1, read the DR register to clear this bit

EV4: STOPF = 1, clear this bit by first reading the SR1 register and then writing the CR1 register.

Note:

1) EV1 event pulls down SCL until the end of the corresponding software sequence.

2) EV2 software sequence must be completed before the current byte transfer is completed.

3) After checking the contents of the SR1 register, the user should perform a complete clearing sequence for each flag set that is found. For example, the ADDR and STOPF flags need to use the following sequence:

If ADDR = 1, read SR first, then read SR2,if STOPF = 1, read SR1 first, and then write CR1.

The purpose of this is to ensure that if both ADDR and STOPF are found to be set, they can both be cleared.

26.3.4.3. Close communication

After the last data byte is transmitted, the master generates a stop condition. When the slave detects this condition:

● The hardware sets STOPF, and if the ITEVTEN bit is set, an interrupt is generated.

By first reading SR1 and then writing CR1, the STOPF bit is cleared. (See EV4 in the figure above)

26.3.5. I2C master mode

In Master mode, the I2C interface starts data transfer and generates a clock signal. Serial data transfers always begin with a START condition and end with a STOP condition.

When a START condition is generated on the bus via the START bit, the device enters master mode.

The following is the sequence of operations required for master mode:

- Set the input clock to the module in the I2C_CR2 register to generate the correct timing
- Configure the clock control register
- Configure the rise time register
- Program the I2C_CR1 register to start the peripheral
- Set the START bit in the I2C_CR1 register to 1 to generate a start condition

  The input clock frequency to the I2C module must be at least:
- In standard mode: 2 MHz
- In fast mode: 4 MHz

26.3.5.1. The host generates clock

The CCR register generates high and low levels of SCL with rising or falling edges. Since the slave may stretch the SCL signal, after the rising edge of SCL occurs, the master checks the SCL signal from the bus when the time programmed in the TRISE register arrives.

— If SCL is low, it means that the slave is stretching the SCL bus, and the high-level counter stops counting until SCL is detected high. This is to ensure a minimum high time for the SCL parameter.

— If SCL is high, the high-level counter keeps counting.

In fact, even if the slave does not stretch SCK, it will take some time for such a feedback loop to occur from the rising edge of SCL to the detection of the rising edge of SCL. The time of this loop is related to the rise time of SCL (VIH data detection of SCL), plus the analog noise filtering of the SCL input path, and the SCL synchronization inside the chip due to the use of the APB clock. The maximum time for the feedback loop is programmed in the TRISE register, so the frequency of SCL remains stable regardless of the rise time of SCL.

26.3.5.2. Start condition

When BUSY = 0, set START = 1, the I2C interface will generate a Start condition and switch to master mode (MSL is set).

Note: Setting the START bit in master mode will generate a ReStart condition by hardware after the current byte is transmitted.

if Start condition is issued:

● The SB bit is set by hardware and an interrupt is generated if the ITEVTEN bit is set.

The master reads the SR1 register, and then writes the slave address to the DR register. (Transfer sequence EV5)

26.3.5.3. Slave address sending

The slave address is sent to the SDA line through the internal shift register.

● In 7-bit address mode, send out an address byte.

If the address byte is sent out

- The ADDR bit is set by hardware and an interrupt is generated if the ITEVTEN bit is set.

Then the Master reads the SR1 register, followed by the SR2 register.

According to the lowest bit of the sent slave address, the master decides to enter the transmitter mode or the receiver mode.

● In 7-bit address mode

- To enter transmitter mode, the master device sets the least significant bit to '0' when sending the slave address.
- To enter receiver mode, the master device sends the slave address with the least significant bit set to '1'.

The TRA bit indicates whether the master is in receiver mode or transmitter mode.

26.3.5.4. Master transmitter

After sending the address and clearing the ADDR bit, the master device sends the byte from the DR register to the SDA line through the internal shift register.

The Master waits until the first data byte is written to the DR register (see EV8_1).

When an ACK pulse is received, the TxE bit is set by hardware and an interrupt is generated if the INEVFEN and ITBUFEN bits are set.

If TxE is set and no new data byte is written to the DR register before the end of the last data transmission, BTF is set by hardware. The I2C interface will keep SCL low until the BTF is cleared (after reading I2C_SR1, then writing the I2C_DR register). Closing Communication.

After writing the last byte in the DR register, a STOP condition is generated by setting the STOP bit (see EV8_2 in Figure), then the I2C interface will automatically return to slave mode (MLS bit cleared).

Note: When the TxE or BTF bit is set, the stop condition should be scheduled on the EV8_2 event.

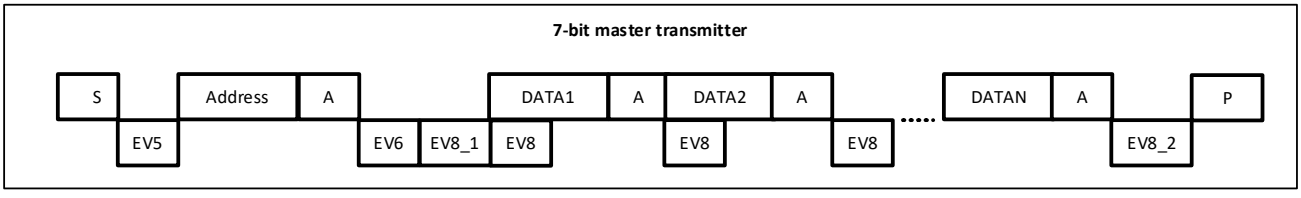

Figure 26-5 Master transmitter transmission sequence diagram

Legend: S = Start, Sr = Repeated Start, P = Stop, A = Acknowledge, EVx = Event(with interrupt if ITEVFEN = 1)

EV5: SB = 1, by reading SR1, and then writing data to the DR register, the bit is cleared

EV6: ADDR = 1, by reading SR1, and then reading SR2, the bit is cleared

EV8_1: TxE = 1, shift register is empty, data register is empty, write Data1 to DR register

EV8: TxE = 1, shift register is not empty, data register is empty, write Data2 to DR register, this bit is cleared

EV8_2: TxE = 1, BTF = 1, Write the Stop bit register, when the hardware sends the Stop bit, TxE and BTF are cleared

Note:

1. EV5, EV6, EV8_1 and EV8_2 events, stretch the low level of SCL until the corresponding software sequence execution ends.
2. EV8 software The sequence must complete before the current byte is sent. If the EV8 software sequence cannot be completed before the end of the currently transmitted byte, it is recommended to use BTF instead of TxE, which has the disadvantage of slowing down the communication.

26.3.5.5. Master receiver

After sending the address and clearing the ADDR, the I2C interface enters the master receiver mode. In this mode, the I2C interface receives data bytes from the SDA line and sends them to the DR register through the internal shift register. After each byte, the I2C interface performs the following operations in sequence:

● If the ACK bit is set, issue an acknowledge pulse.

● The hardware sets RxNE = 1, if the INEVFEN and ITBUFEN bits are set, an interrupt will be generated

If the RxNE bit is set and the data in the DR register is not read before the end of receiving new data, the hardware will set BTF = 1, and the I2C interface will keep SCL low before clearing BTF, after reading I2C_SR1 Reading the I2C_DR register again will clear the BTF bit.

Close communication:

Method 1: The application scenario of this method is: when I2C is set as the highest priority interrupt in the application

The Master sends a NACK after receiving the last byte from the Slave. After receiving the NACK, the Slave releases control of the SCL and SDA lines. The Master can then send a Stop/Restart condition.

1) In order to generate a NACK pulse after the last byte is received, the ACK bit must be cleared after reading the second-to-last data byte (after the second-to-last RxNE event).

2) To generate a STOP/RESTART condition, software must set the STOP/START bit after reading the second-to-last data byte (after the second-to-last RxNE event).

3) When a single byte is received, the closing acknowledge and stop conditions should be generated just after EV6 (EV6_1, after clearing ADDR). After a stop condition is generated, the I2C interface automatically returns to slave mode (MSL bit is cleared).

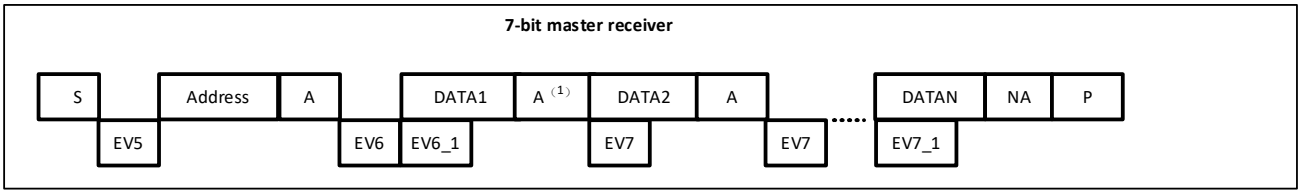

Figure 26-6 Method 1: Timing when master mode transmits

Legend: S = Start, Sr = Repeated Start, P = Stop, A = Acknowledge, EVx = Event(with interrupt if ITEVFEN = 1)

EV5: SB = 1, read SR1, then write DR register, this bit is cleared

EV6:ADDR = 1, read SR1, then read SR2, this bit is cleared

EV6_1: No related flag event, only used for 1 byte reception.

EV7: RxNE = 1, read DR register, this bit is cleared

EV7_1: RxNE = 1, read DR register, write ACK = 0 and set STOP

1) If a single byte is received, the above marked as (1) The place will be NA

2) EV5, EV6 events, stretch the low level of SCL until the corresponding software sequence execution ends

3) EV7 software sequence must be executed before the current byte is sent. In EV7, the software sequence cannot be managed until the currently transferred byte has been transferred. It is recommended to use BTF instead of RXNE, which has the disadvantage of slowing down communication.

4) The software sequence of EV6_1 or EV7_1 must be completed before the ACK of the current byte transmission.

Method 2: The application scenario of this method is: the I2C interrupt is not the highest priority in the application, or the query method.

Use this method, DataN-2 is not read, so after DataN-1, the communication is stretched (RxNE and BTF is set). Then, before reading DataN-2 of the DR register, clear the ACK bit to ensure that it is cleared before DataN ACKs. After this, after reading DataN-2, set the STOP/START bit and read DataN-1. After RxNE is set, read DataN

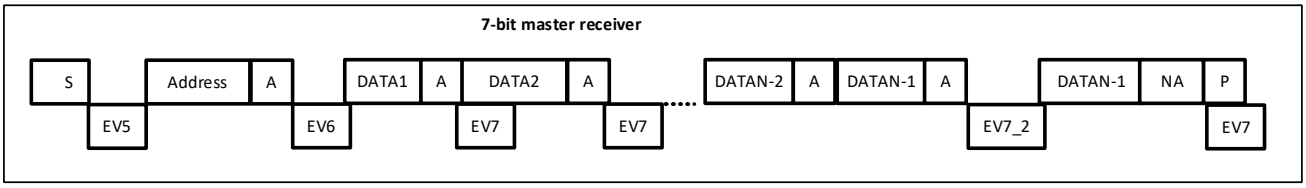

Figure 26-7 Method 2: Timing for master mode transmission when N > 2

Legend: S = Start, Sr = Repeated Start, P = Stop, A = Acknowledge, EVx = Event(with interrupt if ITEVFEN = 1)

EV5: SB = 1, first read SR1 register, then write DR register, clear this bit

EV6: ADDR, first read SR1, then read SR2, clear this bit

EV7: RxNE = 1, read DR register to clear this bit

EV7_2: BTF = 1, DataN-2 is stored in DR register, DataN-1 is stored in shift register, write ACK = 0, read DataN-2 in the DR register. Set STOP, read DataN-1

Note:

1) EV5, EV6 events, stretch the low level of SCL until the corresponding software sequence execution ends

2) EV7 software sequence must be executed before the completion of the current byte transmission. In EV7, the software sequence cannot be managed until the currently transferred byte has been transferred. It is recommended to use BTF instead of RXNE, which has the disadvantage of slowing down communication.

• When 3 bytes are to be read:

— RxNE = 1 = > Nothing (DataN-2 not read).

— DataN-1 received

— BTF = 1, shift and data registers are full: DR register stores DataN-2, shift register stores DataN-1 = > SCL is pulled low: there is no other data to be received on the bus

— Clear the ACK bit

— Read DataN-2 in DR register = > This will start the reception of DataN in shift register

— DataN reception complete (with a NACK)

— Write START or STOP bit

— Read DataN-1

— RxNE = 1

— Read DataN

The above process is a description for N > 2. The reception of 1 byte and 2 bytes uses different processing methods, see the following description:

● In the case of 2 bytes reception

— Set the POS and ACK bits

— Wait for ADDR to be set

— Clear the ADDR bit

— Clear the ACK bit

— Wait for BTF to be set

— Write STOP bit

— Read DR twice

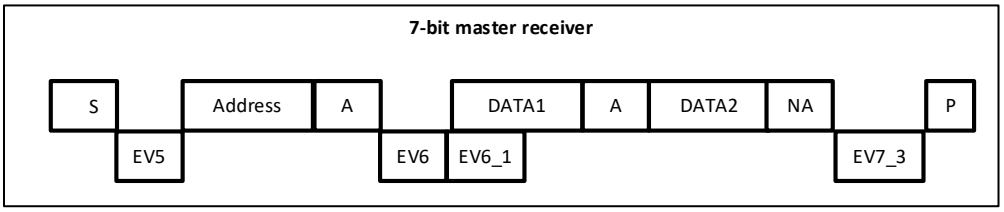

Figure 26-8 Method 2: timing when master mode transmits when N = 2

Legend: S = Start, Sr = Repeated Start, P = Stop, A = Acknowledge, EVx = Event(with interrupt if ITEVFEN = 1)

EV5: SB = 1, first read SR1 register, then write DR register, clear this bit

EV6: ADDR = 1, first read the SR1 register, then read the SR2 register, clear the ADDR bit

EV6_1: no related flag events. After EV6, that is, after the address is cleared, ACK should be cleared

EV7_3: BTF = 1, write STOP = 1, then read DR twice (Data1 and Data2)

Note: 1) EV5, EV6 events, stretch SCL 2) The software sequence of EV6_1 must be completed before the ACK of the current byte transmission

● In the case of single byte reception

— In ADDR event, clear the ACK bit

— Clear ADDR

— Write STOP or START bit

— Read data after RxNE flag is set

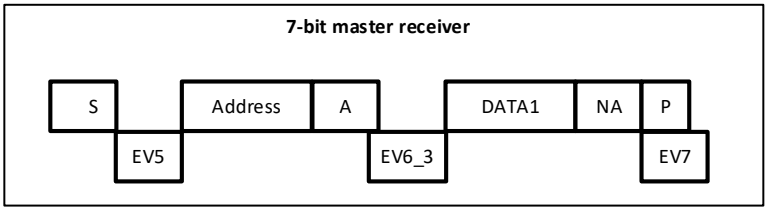

Figure 26-9 Method 2: timing when master mode transmits when N = 1

Legend: S = Start, Sr = Repeated Start, P = Stop, A = Acknowledge, EVx = Event(with interrupt if ITEVFEN = 1)

EV5: SB = 1, first read SR1 register, then write DR register, clear this bit

EV6_3: ADDR = 1, write ACK = 0. First read the SR1 register, then read the SR2 register, clear the ADDR bit. After ADDR is cleared, write STOP = 1

EV7: RxNE = 1, read DR register to clear this bit

Note:

EV5 event will stretch the low level of SCL until the corresponding software sequence execution ends.

26.3.6. Error stage

26.3.6.1. Bus error(BERR)

A bus error is generated when the I2C interface detects an external stop or start condition during an address or data byte transfer. at this time:

● The BERR bit is set to '1', if the ITERREN bit is set, an interrupt is generated

● In slave mode: the data is discarded, the hardware releases the bus:

─ If it is a wrong Start condition, the slave considers a Restart and waits for an address or a stop condition

─ If it is a wrong Stop condition, the slave operates according to the normal stop condition, and the hardware releases the bus at the same time

● In master mode: the hardware does not release the bus, and does not affect the current transmission status. At this point it is up to the software to decide whether to abort the current transfer.

26.3.6.2. ACK Failed(AF)

An acknowledge error occurs when the interface detects a no acknowledge bit. at this time:

● The AF bit is set, and an interrupt is generated if the ITERREN bit is set

● When the transmitter receives a NACK, the communication must be reset:

─ If it is in slave mode, the hardware releases the bus.

─ If in master mode, software must generate a stop condition or repeated start.

26.3.6.3. Arbitration loss (ARLO)

Arbitration loss error occurs when I2C interface detects arbitration loss, at this time:

● The ARLO bit is set by hardware and an interrupt is generated if the ITERREN bit is set

● The I2C interface automatically returns to slave mode (MSL bit is cleared). When the I2C interface loses arbitration, it cannot respond to its slave address in the same transfer, but it can respond after the master that wins the bus sends a repeated start condition

● Hardware release bus

26.3.6.4. Overrun/Underrun error (OVR)

In slave mode, if the clock extension is disabled and the I2C interface is receiving data, when it has received a byte (RxNE = 1), but the previous byte data in the DR register has not been read, an overrun occurs. mistake. at this time:

● The last received data is discarded

● On overrun error, software should clear the RxNE bit and the transmitter should resend the last transmitted byte

In slave mode, if the clock extension is disabled and the I2C interface is sending data, before the clock of the next byte arrives, the new data has not been written to the DR register (TxE = 1), an underrun error occurs. at this time:

● The previous byte in the DR register will be repeated

● The user should make sure that when an underrun error occurs, the receiver should discard the repeatedly received data. The transmitter should update the DR register at the specified time according to the I2C bus standard

When sending the first byte, the DR register must be written to after clearing ADDR and before the first rising edge of SCL, if this is not possible, the receiver should discard the first data.

26.3.7. SDA/SCL control

● If clock stretching is allowed:

─ Transmitter mode: If TxE = 1 and BTF = 1: The I2C interface keeps the clock line low before transmission, waiting for software to read SR1, and then write the data into the data register (DR and shift registers are both empty).

─ Receiver mode: if RxNE = 1 and BTF = 1: I2C interface keeps clock line low after receiving data byte, waiting for software to read SR1, then read data register DR (DR and shift registers are both full).

● If clock stretching is disabled in slave mode:

─ If RxNE = 1, the DR has not been read before the next byte is received, an overun occurs. The last byte received is lost.

─ If TxE = 1, an underrun occurs when no new data is written into the DR before the next byte must be sent. The same bytes will be sent repeatedly.

─ Hardware does not implement control of write collisions.

26.3.8. DMA requests

DMA requests (when enabled) are only used for data transfers. When the data register becomes empty when sending, or the data register becomes full when receiving, a DMA request is generated. DMA must be initialized and enabled before the end of the current byte transfer. The DMAEN bit (in the I2C_CR2 register) must be enabled before an ADDR event occurs.

In master or slave mode, when clock stretching is enabled, the DMAEN bit can be set during an ADDR event before clearing ADDR. A DMA request must be responded to before the current byte transfer is complete. When the length of the DMA transfer data reaches the value set by the DMA, the DMA controller sends EOT (End of transfer) to the I2C, and generates a transfer complete interrupt (if the interrupt enable bit is valid):

● Master transmitter: In the EOT interrupt service routine, the DMA request needs to be disabled, and then the stop condition is set after waiting for the BTF event.

● Master receiver: When the number of data to be received is greater than or equal to 2, the DMA controller sends a hardware signal EOT_1, which corresponds to DMA transmission (byte number - 1). If the LAST bit is set in the I2C_CR2 register, the hardware will automatically send a NACK after sending the next byte after EOT_1. With the interrupt enabled, the user can generate a stop condition in the interrupt service routine when the DMA transfer is complete.

26.3.8.1. Transmission using DMA

DMA mode can be enabled by setting the DMAEN bit in the I2C_CR2 register. As long as the TxE bit is set, the data will be loaded into the I2C_DR register from the preset memory area by DMA. To assign a DMA channel to I2C, perform the following steps (x is the channel number):

1. Set the I2C_DR register address in the DMA_CPARx register. Data will be transferred from memory to this address after each TxE event.
2. Set the memory address in the DMA_CMARx registers. Data is transferred from this bank to I2C_DR after each TxE event.
3. Set the desired number of bytes to transfer in the DMA_CNDTRx registers. This value will be decremented after each TxE event.
4. Configure the channel priority using the PL[0:1] bits in the DMA_CCRx register.
5. Set the DIR bit in the DMA_CCRx register, and can configure the interrupt request when the entire transfer is half or fully completed according to the application requirements.
6. Activate the channel by setting the EN bit on the DMA_CCTx register.

When the number of data transfers set in the DMA controller has been completed, the DMA controller sends an EOT/EOT_1 signal that the transfer is over to the I2C interface. When interrupts are enabled, a DMA interrupt will be generated.

Note: Do not set the ITBUFEN bit in the I2C_CR2 register if using DMA for transmission.

26.3.8.2. Reception using DMA

The DMA receive mode can be activated by setting the DMAEN bit in the I2C_CR2 register. Each time a data byte is received, the data in the I2C_DR register will be transferred to the set memory area by the DMA (refer to the DMA description). To set up a DMA channel for I2C reception, perform the following steps (x is the channel number):

1. Set the address of the I2C_DR register in the DMA_CPARx register. Data will be transferred from this address to the memory area after each RxNE event.
2. Set the bank address in the DMA_CMARx registers. Data will be transferred from the I2C_DR register to this bank after each RxNE event.
3. Set the desired number of bytes to transfer in the DMA_CNDTRx registers. This value will be decremented after each RxNE event.
4. Configure the channel priority using PL[0:1] in the DMA_CCRx register.
5. Clear the DIR bit in the DMA_CCRx register. According to the application requirements, it can be set to issue an interrupt request when half or all of the data transfer is completed.
6. Set the EN bit in the DMA_CCRx register to activate the channel.

When the number of data transfers set in the DMA controller has been completed, the DMA controller sends an EOT/EOT_1 signal that the transfer is over to the I2C interface. When interrupts are enabled, a DMA interrupt will be generated.

Note: If using DMA for reception, do not set the I2C_CR2 register's ITBUFEN bit.

26.4. I2C interrupts

Table 26-1 I2C interrupt requests

Interrupt event	Event flag	Interrupt enable control bit
START has been sent (Master)	SB	ITEVTEN
Address has been sent(Master) / Address matched(Slave)	ADDR	
Stop detection interrupt flag(Slave)	STOPF	
Transfer Complete Reload	BTF	
Receive buffer not empty	RxNE	ITEVTEN and ITBUFEN
Transmit buffer empty	TxE	
Bus error	BERR	ITERREN
Arbitration loss(Master)	ARLO	
ACK Failed	AF	
Overrun/Underrun	OVR	
PEC error	PECERR	

26.5. I2C registers

Registers can be accessed half-word or word.

26.5.1. I2C control register 1 (I2C_CR1)

Address offset: 0x00

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SWRST	Res	Res	Res	POS	ACK	STOP	START	NOSTRETCH	ENGC	Res	Res	Res	Res	Res	PE
RW				RW	RW	RW	RW	RW	RW						RW

Bit	Name	R/W	Reset Value	Function
15	SWRST	RW	0	Software reset. When set, I2C is in reset state. Before the reset release, make sure that the I2C pins are released and the bus is in an idle state. 0: I2C module is not in reset state 1: I2C module is in reset state Note: This bit can be used to reinitialize I2C in error or locked state. If the BUSY bit is 1, no stop condition is detected on the bus.
14:12	Reserved	RES	-	Reserved
11	POS	RW	0	ACK/PEC position (for data reception), this register can be set/cleared by software, or cleared by hardware when PE = 0. 0: The ACK bit controls the (N)ACK of the byte currently being received in the shift register. The PEC bit indicates that the byte in the current shift register is PEC 1: The ACK bit controls the (N)ACK of the next byte received in the shift register. The PEC bit indicates that the next byte received in the shift register is a PEC Note: The POS bit can only be used in a 2-byte receive configuration and must be configured before receiving data. In order to NACK the 2nd byte, the ACK bit must be cleared after clearing ADDR. In order to detect the PEC of the second byte, the PEC bit must be set during the ADDR stretch event after the POS bit is configured.
10	ACK	RW	0	acknowledgment enable. This register can be set/cleared by software, or cleared by hardware when PE = 0. 0: no response returned

				1: Return an acknowledgment after a byte has been received. (matching address or data)
9	STOP	RW	0	When a stop condition occurs, software can set/clear this register, or when a stop condition is detected, it is cleared by hardware, when a timeout error is detected, hardware is set. In Master Mode: 0: No stop generation 1: Generate a stop condition after the current byte transfer or after the current start condition is issued In Slave Mode: 0: No stop condition is generated 1: Release the SCL and SDA lines after the current byte transfer
8	START	RW	0	The starting condition is generated. This register can be set/cleared by software, or cleared by hardware when a START condition is issued or when PE = 0. Master mode: 0: No start generation 1: Restart/Start condition Slave Mode: 0: No start generation 1: When the bus is idle, generate a start generation (and automatically switch to Master Mode by hardware)
7	NOSTRETCH	RW	0	Clock stretching (Slave) is disabled. When the ADDR or BTF flag is set, this bit is used by the slave to disable clock stretching until reset by software. 0: allow clock stretching 1: Disable clock stretching
6	ENGC	RW	0	General call enabled. 0: Disable general call. Respond to address 00h with NACK 1: Allow general calls. Responds to address 00h with an ACK
5:1	Reserved	RES	-	Reserved
0	PE	RW	0	I2C module enabled. 0: Disable 1: I2C enable Note: If a communication is in progress when this bit is cleared, after the current communication ends, the I2C module is disabled and returns to the idle state. Since PE = 0 after the communication ends, all bits are cleared. In master mode, this bit must never be cleared until the communication has ended.

Note: When the STOP/START/PEC bit is set, software should not perform any write operations to I2C_CR1 until the hardware clears this bit, otheRWise, the STOP/START/PEC bit may be set a second time.

26.5.2. I2C control register 2 (I2C_CR2)

Address offset: 0x04

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	LAST	DMAEN	IT-BUFEN	ITEV-TEN	ITER-REN	Res	Res	FREQ[5:0]					
			RW	RW	RW	RW	RW			RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
15:13	Reserved	RES	-	Reserved
12	LAST	RW	0	DMA last transfer. 0: EOT of next DMA is not the last transfer

				1: The EOT of the next DMA is the last transfer Note: This bit is used in master receive mode to allow a NACK to be generated on the last received data.
11	DMAEN	RW	0	DMA request enabled. 0: Disable DMA request, 1: DMA requests are enabled when TxE = 1 or RxNE = 1.
10	ITBUFEN	RW	0	Buffer interrupt enable. 0: No interrupt is generated when TxE = 1 or RxNE = 1 1: When TxE = 1 or RxNE = 1, an event interrupt is generated (regardless of the value of DMAEN)
9	ITEVTEN	RW	0	Event interrupt enable. 0: Disable 1: Enable event interrupt This interrupt will be generated under thefollowing conditions: ● SB = 1 (main mode), ● ADDR = 1 (Master/Slave mode) ● STOPF = 1 (slave mode) ● BTF = 1, but no TxE or RxNE events ● If ITBUFFEN = 1, TxE event is 1 ● If ITBUFEN = 1, the RxNE event is 1
8	ITERREN	RW	0	Error interrupt enable. 0: Disable error interrupt, 1: Enable error interrupt, This interrupt will be generated under the following conditions: ● BERR = 1 ● ARLO = 1 ● AF = 1 ● OVR = 1 ● PECERR = 1
7:6	Reserved	RES	-	Reserved
5:0	FREQ	RW	0	I2C module clock frequency. This register must be configured with the value of the APB clock frequency to generate data setup and hold times that are compatible with the I2C protocol. The minimum allowable frequency that can be set is 4 MHz (standard mode, ie 100 k), 12 MHz (400 k), and the maximum frequency is the highest APB clock frequency of the chip. 000000: Forbidden 000001: Forbidden 000100: 4 MHz … 100100: 36 MHz … 110000: 48 MHz Greater than 100100: Forbidden.

26.5.3. I2C own address register 1 (I2C_OAR1)

Address offset: 0x08

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	Res	Res	Res		ADD[7:1]							Res
									RW	RW	RW	RW	RW	RW	

Bit	Name	R/W	Reset Value	Function
14:8	Reserved	RES	-	Reserved
7:1	ADD[7:1]	RW	0	Bit 7:1 of address
0	reserved			

26.5.4. I2C data register (I2C_DR)

Address offset: 0x10

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	Res	Res	Res	Res	DR[7:0]							
								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
15:8	Reserved	RES	-	Reserved
7:0	DR[7:0]	RW	0	8-bit data register, two independent buffers inside the chip share an address, which are used to store the received data (RX_DR) and place the data to be sent to the bus (TX_DR). Transmitter Mode: Data transfer is automatically started when a byte is written to the DR register (actually written to TX_DR). Once the transmission starts (TxE = 1), if the next data to be transmitted can be written into the DR register in time, the I2C module will maintain a continuous data flow. Receiver Mode: The received byte is copied to the DR register (actually RX_DR) (RxNE = 1). Read out the data register before receiving the next byte (RxNE = 1) to realize continuous data reception. Note: 1) In slave mode, the address will not be copied into the data register DR 2) The hardware does not handle write conflicts (if TxE = 0, the data register can still be written) 3) If the ARLO event occurs while processing the ACK pulse, the received byte will not be copied to the data register, so it cannot be read

26.5.5. I2C stage register 1 (I2C_SR1)

Address offset: 0x14

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	PECERR	OVR	AF	ARLO	BERR	TxE	RxNE	Res	STOPF	Res	BTF	ADDR	SB
			RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	R	R		R		R	R	R

Bit	Name	R/W	Reset Value	Function
15:13	Reserved	RES	-	Reserved
12	PECERR	RC_W0	0	A PEC error occurred while receiving. 0: No PEC error, return ACK after receiving PEC (if ACK = 1) 1: There is a PEC error, and NACK is returned after receiving PEC (regardless of the value of ACK) This bit is cleared by software by writing 0, or by hardware when PE = 0.
11	OVR	RC_W0	0	Overload/Underload flag. 0: no overload/underload, 1: Overload/underload occurred. When NOSTRETCH = 1, this bit is set by hardware in slave mode, In the receive mode, when a new byte is received (including the ACK response pulse), and the contents of the data register have not been read out, the newly received byte will be lost. In transmit mode when a new byte is to be sent, but no new data is written to the data register, the same byte will be sent twice.

				This bit is cleared by software by writing 0, or by hardware when PE = 0. Note: If a write to the data register occurs very close to the rising edge of SCL, the transmitted data is indeterminate and a hold time error occurs.
10	AF	RC_W0	0	Reply failure flag. 0: no response failed, 1: Reply failed. Hardware will set this register when no acknowledgement is returned. This bit is cleared by software by writing 0, or by hardware when PE = 0.
9	ARLO	RC_W0	0	Arbitration lost (main mode). 0: no arbitration loss is detected. 1: Arbitration loss detected. This register is set by hardware when the interface loses control of the bus to another host. This bit is cleared by software by writing 0, or by hardware when PE = 0. After an ARLO event, the I2C interface automatically switches back to slave mode (M/SL = 0).
8	BERR	RC_W0	0	Bus error flag. 0: No start or stop condition error. 1: Error in start or stop condition. This bit is set by hardware when the interface detects a false start or stop condition. This bit is cleared by software by writing 0, or by hardware when PE = 0.
7	TxE	R	0	The data register is empty (when transmitting) flag. 0: The data register is not empty. 1: The data register is empty. When sending data, this bit is set to 1 when the data register is empty, and this bit is not set during the sending address phase. This bit can be cleared by software writing data to the DR register, or automatically by hardware after a START or STOP condition occurs, or when PE = 0. This bit is not set if a NACK is received, or if the next byte to be sent is PEC (PEC = 1). Note: After writing the first data to be sent, or writing data when BTF is set, the TxE bit cannot be cleared, because the data register is empty at this time.
6	RxNE	R	0	Data register not empty (on reception) flag. 0: The data register is empty. 1: The data register is not empty. On reception, this register is set when the data register is not empty. During the receive address phase, this register is not set. This register is cleared by software reads and writes to the data register, or by hardware when PE = 0. Note: When BTF is set, reading data does not clear the RxNE bit because the data register is still full.
5	Reserved	RES	-	Reserved
4	STOPF	R	0	Stop condition detection bit (slave mode). 0: No stop bit detected. 1: Stop condition detected. After an acknowledgment (if ACK = 1), the hardware sets this bit to 1 when the slave device detects a stop condition on the bus. After the software reads the I2C_SR1 register, a write to the I2C_CR1 register will clear this bit, or when PE = 0, the hardware will clear this bit. Note: The STOPF bit is not set after a NACK is received.

3	Reserved			
2	BTF	R	0	End of byte transfer flag. 0: Byte transfer not complete 1: Byte transfer ended successfully The hardware will set this register in the following cases (when slave mode, NOSTRETCH = 0, master mode, regardless of NOSTRETCH): — On reception, when a new byte is received (including the ACK pulse) and the data register has not been read (RxNE = 1). — When transmitting, when a new data should be transmitted and the data register has not been written with new data (TxE = 1). This bit is cleared by a read or write to the data register after software reads the I2C_SR1 register, or by hardware after sending a start or stop condition, or when PE = 0. Note: After receiving a NACK, the BTF bit is not set.
1	ADDR	R	0	Address has been sent (Master mode)/Address matched (Slave mode). After the software reads the I2C_SR1 register, reading the I2C_SR2 register will clear this bit, when PE = 0, it will be cleared by hardware. Address matching (Slave): 0: The address does not match or the address was not received, 1: The received address matches. Hardware will set this bit when the received slave address matches the OAR register or general call address. Note: In slave mode, it is recommended to perform a complete clearing sequence, that is, after ADDR is set, read the SR1 register first, and then read the SR2 register. Address has been sent (Master): 0: Address sending has not ended, 1: Address sending ends. For a 7-bit address, it is set when the ACK byte is received. Note: This register will not be set after receiving a NACK.
0	SB	R	0	Start bit flag (master mode). 0: start condition not sent, 1: The start condition has been sent, — This register is set when a START condition is sent. — After the software reads the I2C_SR1 register, a write to the data register will clear this bit, or when PE = 0, it will be cleared by hardware.

26.5.6. I2C stage registor 2 (I2C_SR2)

Address offset: 0x18

Reset value: 0x0000

Note: Even if the ADDR flag is set after reading the I2C_SR1 register, reading the I2C_SR2 register after reading I2C_SR1 will clear the ADDR flag. Therefore, the I2C_SR2 register must be read only when the ADDR bit of the I2C_SR1 register is found to be set or the STOPF bit is cleared.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PEC[7:0]								Res	Res	Res	GEN-CALL	Res	TRA	BUSY	MSL
R	R	R	R	R	R	R	R				R		R	R	R

Bit	Name	R/W	Reset Value	Function
15:7	Reserved			

4	GENCALL	R	0	General call address (slave mode). 0: No broadcast call address received, 1: When ENGC = 1, the address of the general call is received. Hardware clears this register when a STOP condition or a repeated START condition occurs, or when PE = 0.
3	Reserved	RES	-	Reserved
2	TRA	R	0	send/receive flag. 0: data received 1: Data has been sent At the end of the entire address transfer phase, this register is set according to the R/W bit of the address byte. Hardware clears this register when a STOP condition is detected (STOPF = 1), or a repeated-START condition, or bus arbitration is lost (ARLO = 1), or when PE = 0.
1	BUSY	R	0	Bus busy flag. 0: No data communication on the bus 1: Polarity data communication is in progress on the bus Set by hardware when SDA or SCL is detected low. Hardware clears when a stop condition is detected. This register indicates the current bus communication in progress, this information is still updated when the interface is disabled (PE = 0).
0	MSL	R	0	Master-slave mode. 0: slave 1: master — Set by hardware when the interface is in master mode (SB = 1), — Hardware cleared when a stop condition is detected on the bus (STOPF = 1), arbitration lost (ARLO = 1), or when PE = 0.

26.5.7. I2C clock control register (I2C_CCR)

Address offset: 0x1C

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
F/S	DUTY	Res	Res	CCR[11:0]											
RW	RW			RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
15	F/S	RW	0	I2C Master mode selection. 0: Standard mode 1: Fast Mode
14	DUTY	RW	0	Duty cycle in fast mode. 0: In fast mode: T_{low}/T_{high} = 2 1: In fast mode: T_{low}/T_{high} = 16/9
13:12	Reserved	RES	-	Reserved
11:0	CCR[11:0]	RW	0	Clock control division factor in fast/standard mode (master mode). This division factor is used to set the SCL clock in master mode. ● Standard Mode: ✓ T_{high} = CCR x Tpclk ✓ T_{low} = CCR x Tpclk ● Express Mode: ✓ DUTY = 0: T_{high} = CCR x Tpclk Tlow = 2 x CCR x Tpclk ✓ DUTY = 1 (to reach 400KHz): T_{high} = 9 x CCR x Tpclk

				Tlow = 16 x $_{CCR}$ x Tpclk Note: 1. The minimum allowed setting is 0x04, and the minimum allowed in fast DUTY mode is 0x01 2. $T_{high} = t_{r(SCL)} + t_{w(SCLH)}$ 3. $T_{low} = t_{r(SCL)} + t_{w(SCLL)}$ 4. These delays have no filter 5. This register can only be configured when PE = 0, 6. f_{CK} should be an integer multiple of 10 MHz, so that the fast 400 kHz can be correctly generated at which

26.5.8. I2C TRISE register (I2C_TRISE)

Address offset: 0x20

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	TRISE[5:0]					
										RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
15:6	Reserved	RES	-	Reserved
5:0	TRISE	RW	0	Maximum rise time in fast/standard mode (master mode). These bits should provide the maximum duration of the S CL feedback loop in master mode. The purpose of this is to maintain a stable frequency of SCL regardless of the duration of the rising edge of SCL. These bits must be set to the maximum SCL rise time given in the I2C bus specification in 1 increments. For example: the maximum allowable SCL rise time in standard mode is 1000ns. If the value in FREQ [5:0] in the I2C_CR2 register is equal to 0x08, Tpclk = 125ns, then TRISE is configured as 0x09 (1000ns/125ns = 8 + 1 = 9). Filter values can also be added to TRISE. If the result is not an integer, the integer part is written to TRISE to ensure the t_{HIGH} parameter. Note: This register can only be set when PE = 0.

26.5.9. I2C register map

Offset	Register	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	I2C_CR1	SWRST	Res.	Res.	Res.	POS	ACK	STOP	START	NOS-TRETCH	ENGC	ENPEC	Res.	Res.	Res.	Res.	PE
	Reset value	0				0	0	0	0	0	0	0					0
0x04	I2C_CR2	Res.	Res.	Res.	LAST	DMAEN	IT-BUFEN	ITEV-TEN	ITER-REN	Res.	Res.	FREQ[5:0]					
	Reset value				0	0	0	0	0			0	0	0	0	0	0
0x08	I2C_OAR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ADD[7:1]							Res.
	Reset value									0	0	0	0	0	0	0	
0x10	I2C_DR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DR[7:0]							
	Reset value									0	0	0	0	0	0	0	0
0x14	I2C_SR1	Res.	Res.	Res.	PECERR	OVR	AF	ARLO	BERR	TXE	RXNE	Res.	STOPF	Res.	BTF	ADDR	SB
	Reset value				0	0	0	0	0	0	0		0		0		0

0x18	I2C_SR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	GEN-CALL	Res.	TRA	BUSY	MSL
	Reset value												0		0	0	0
0x1C	I2C_CCR	F/S	DUTY	Res.	Res.	CCR[11:0]											
	Reset value	0	0			0	0	0	0	0	0	0	0	0	0	0	0
0x20	I2C_TRISE	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TRISE[5:0]					
												0	0	0	0	1	0

27. Universal synchronous asynchronous receiver transmitter (USART)

27.1. Introduction

The universal synchronous asynchronous receiver transmitter (USART) offers a flexible means of Full-duplex data exchange with external equipment requiring an industry standard NRZ asynchronous serial data format. The USART offers a very wide range of baud rates using a programmable baud rate generator.

It supports synchronous one-way communication and Half-duplex Single-wire communication, as well as multi-processor communications.

High speed data communication is possible by using the DMA (direct memory access) for multibuffer configuration.

27.2. USART main features

- Full-duplex asynchronous communications
- NRZ standard format
- Configurable oversampling method by 16 or 8 to give flexibility between speed and clock tolerance
- A common programmable transmit and receive baud rate of up to 4.5Mbit/s.
- Programmable data word length (8 or 9 bits)
- Configurable stop bits (1 or 2 stop bits)
- Synchronous mode and clock output for synchronous communications
- Single-wire Half-duplex communications
- Separate enable bits for transmitter and receiver
- Hardware flow control
- Receive/transmit bytes by DMA buffer
- Detection flag
  - Receive buffer full
  - Send buffer empty
  - End of transmission
- Parity Control
  - Send check digit
  - Check the received data
- Flagged interrupt sources
  - CTS change
  - Send register empty
  - Send completed
  - Receive data register full
  - Bus idle detected
  - Overflow error

- Frame error
- Noise operation
- Error detection

- Multiprocessor communication
  - If the address does not match, enter silent mode
- Wake-up from silent mode: description by idle detection and address flag USART functional.

27.3. USART function description

USART interface is connected with other devices through three pins. Any USART bidirectional communication requires a minimum of two pins: Receive data In (RX) and Transmit data Out (TX):

RX: Receive data Input. This is the serial data input. Oversampling techniques are used for data recovery by discriminating between valid incoming data and noise.

TX: Transmit data Output. When the transmitter is disabled, the output pin returns to its I/O port configuration. When the transmitter is enabled and nothing is to be transmitted, the TX pin is at high level. In Single-wire and Smartcard modes, this I/O is used to transmit and receive the data.

- An Idle Line prior to transmission or reception
- A start bit
- A data word (8 or 9 bits) least significant bit first
- 1, 2 stop bits, thus indicating the end of the data frame
- The USART interface uses a baud rate generator: Representation of 12-bit Integers and 4-bit Fraction
- A status register (USART_SR)
- Data registers (USART_DR)
- A baud rate register (USART_BRR)

The following pins are required in synchronous mode:

CK: Transmitter clock output.

Clock output. This pin outputs the transmitter data clock for synchronous transmission corresponding to SPI master mode (no clock pulses on start bit and stop it, and a software option to end a clock pulse on the last data bit). In parallel, data an be received synchronously on RX. This can be used to control peripherals that ave shift registers (e.g. LCD drivers). The clock phase and polarity are software rogrammable.

The following pins are required in RS232 Hardware flow control mode:

- **nCTS**: Clear To Send blocks the data transmission at the end of the current transfer when high
- **nRTS**: Request to send indicates that the USART is ready to receive data (when low).

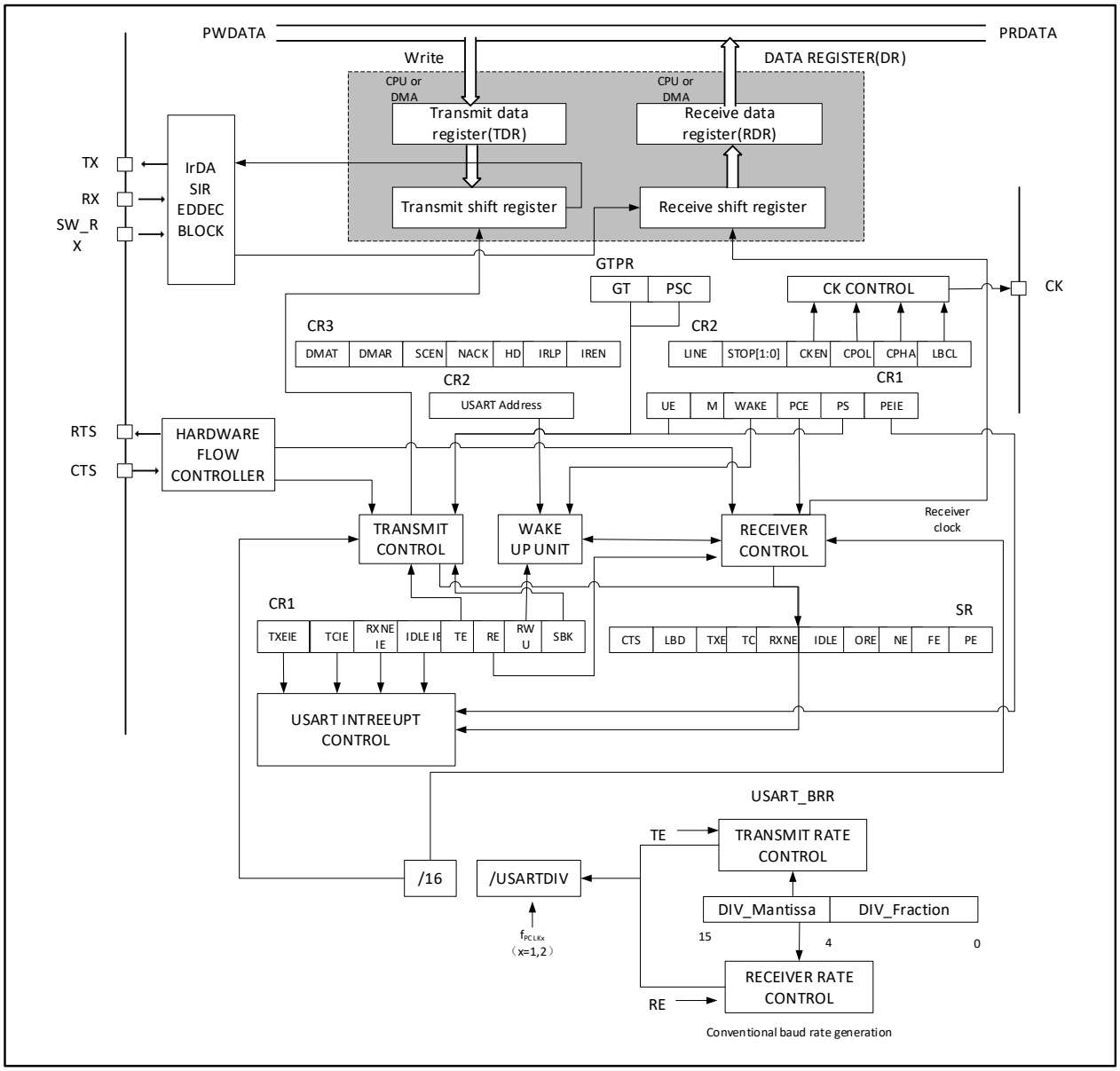

Figure 27-1 USART block diagram

27.3.1. USART character description

World length may be selected as being either 8 or 9 bits by programming the M bits in the USART_CR1 register. The TX pin is low during the start bit and high during the stop bit.

An **Idle character** is interpreted as an entire frame of “1”s followed by the start bit of the next frame containing the data(the number of “1”s includes the number of stop bits)

A **Break character** is interpreted on receiving “0”s for a frame period(including the stop bit period, which is also '0'). At the end of the break frame, the transmitter inserts 1 or 2 stop bits.

Transmission and reception are driven by a common baud rate generator, the clock for each is generated when the enable bit is set respectively for the transmitter and receiver.

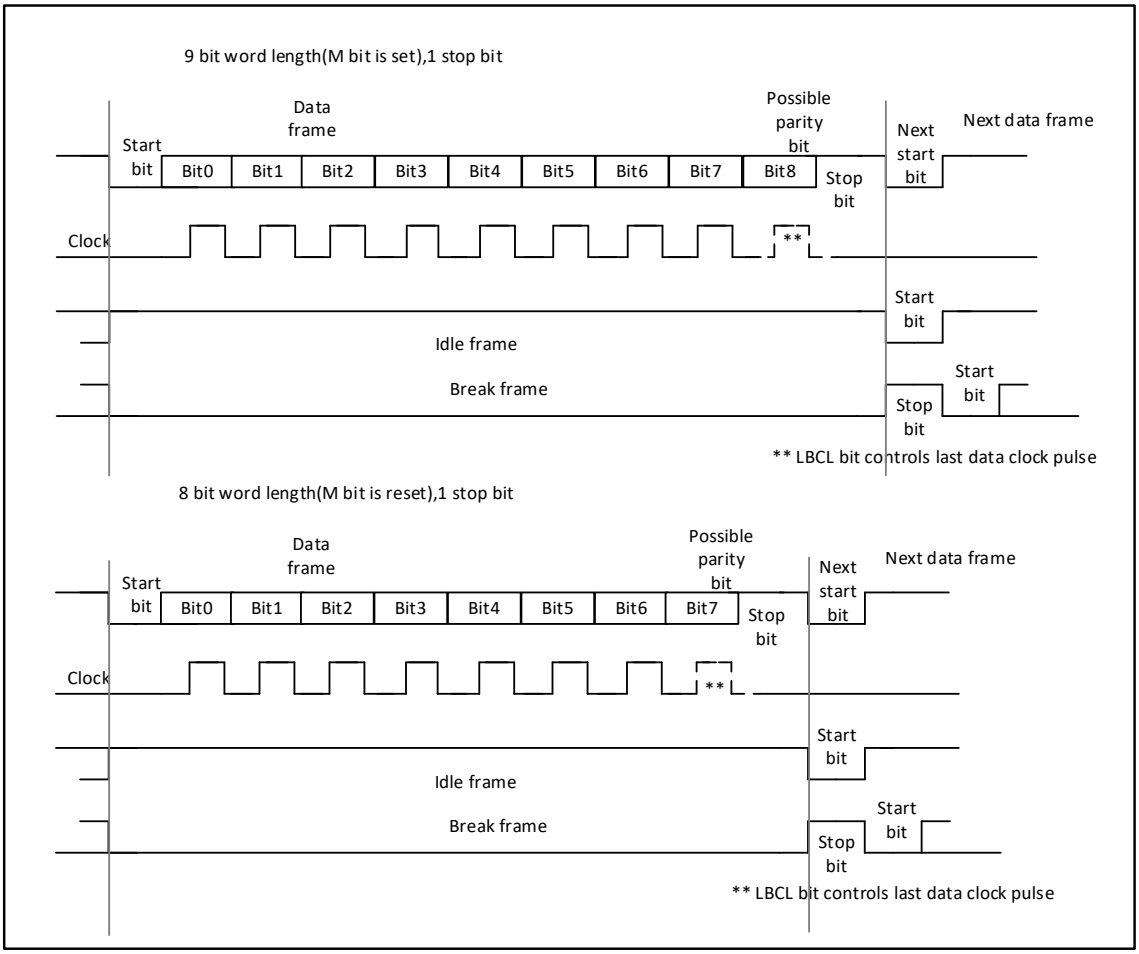

Figure 27-2 Word length programming

27.3.2. Transmitter

The transmitter can send data words of either 8 or 9 bits depending on the M bits status. The Transmit Enable bit (TE) must be set in order to activate the transmitter function. The data in the transmit shift register is output on the TX pin and the corresponding clock pulses are output on the CK pin.

27.3.2.1. Character transmission

During an USART transmission, data shifts out least significant bit first (default configuration) on the TX pin. In this mode, the USART_DR register consists of a buffer (DR) between the internal bus and the transmit shift register.

Every character is preceded by a start bit which is a logic level low for one bit period. The character is terminated by a configurable number of stop bits. The USART supports multiple stop bit configurations: 1 and 2 stop bits.

Note:

The TE bit should not be reset during transmission of data. Resetting the TE bit during the transmission will corrupt the data on the TX pin as the baud rate counters will get frozen. The current data being transmitted will be lost.

An idle frame will be sent after the TE bit is enabled.

27.3.2.2. Configurable stop bits

The number of stop bits to be transmitted with every character can be programmed in Control register 2, bits 13,12.

1) 1 stop bit: This is the default value of number of stop bits.

2) 2 stop bits: This will be supported by normal USART, Single-wire and Modem modes. An idle frame transmission will include the stop bits.

A break transmission will be 10 low bits (when m = 0) or 11 low bits (when m = 1) followed by 2 stop bits. It is not possible to transmit long breaks (break of length greater than 9/10 low bits).

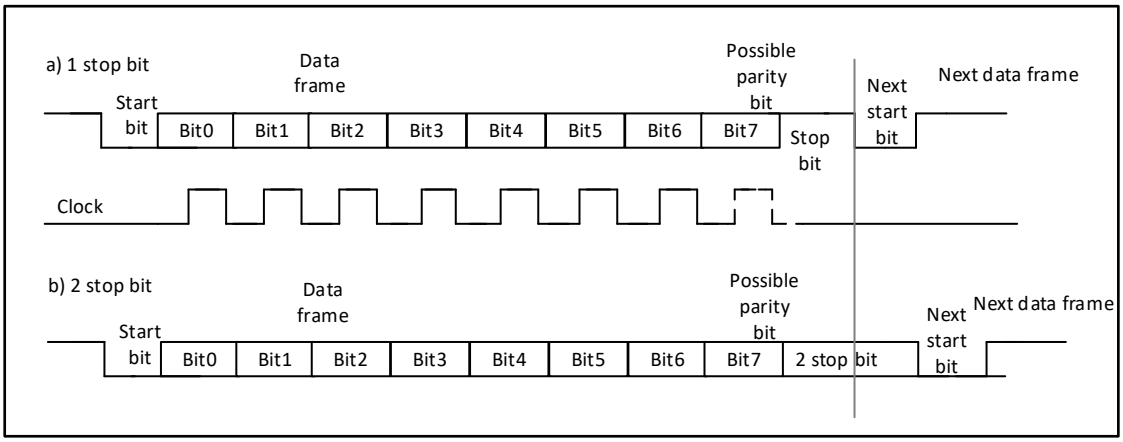

Figure 27-3 Configurable stop bits

Character transmission procedure

1) Enable the USART by writing the UE bit in USART_CR1 register to 1.

2) Program the M bits in USART_CR1 to define the word length.

3) Program the number of stop bits in USART_CR2.

4) Select DMA enable (DMAT) in USART_CR3 if multibuffer communication is to take place. Configure the DMA register as explained in multibuffer communication.

5) Select the desired baud rate using the USART_BRR register.

6) Set the TE bit in USART_CR1 to send an idle frame as first transmission.

7) Write the data to send in the USART_DR register (this clears the TXE bit). Repeat this for each data to be transmitted in case of single buffer.

8) After writing the last data into the USART_DR register, wait until TC = 1. This indicates that the transmission of the last frame is complete. This is required for instance when the USART is disabled or enters the Halt mode to avoid corrupting the last transmission.

27.3.2.3. Single byte communication

Clearing the TXE bit is always performed by a write to the transmit data register. The TXE bit is set by hardware and it indicates:

- The data has been moved from the USART_TDR register to the shift register and the data transmission has started.
- The USART_TDR register is empty.

- The next data can be written in the USART_TDR register without oveRWriting the previous data.

This flag generates an interrupt if the TXEIE bit is set.

When a transmission is taking place, a write instruction to the USART_DR register stores the data in the DR register, next, the data is copied in the shift register at the end of the currently ongoing transmission.

When no transmission is taking place, a write instruction to the USART_DR register places the data in the shift register, the data transmission starts, and the TXE bit is set.

If a frame is transmitted (after the stop bit) and the TXE bit is set, the TC bit goes high. An interrupt is generated if the TCIE bit is set in the USART_CR1 register.

After writing the last data in the USART_TDR register, it is mandatory to wait for TC = 1 before disabling the USART or causing the microcontroller to enter the low-power mode

Use the following software procedure to clear the TC bit:

1. Read the USART_SR register once,
2. Write the USART_DR register once.

Note: The TC bit can also be cleared by software by writing '0' to it. This clearing method is only recommended for use in multi-buffer communication mode.

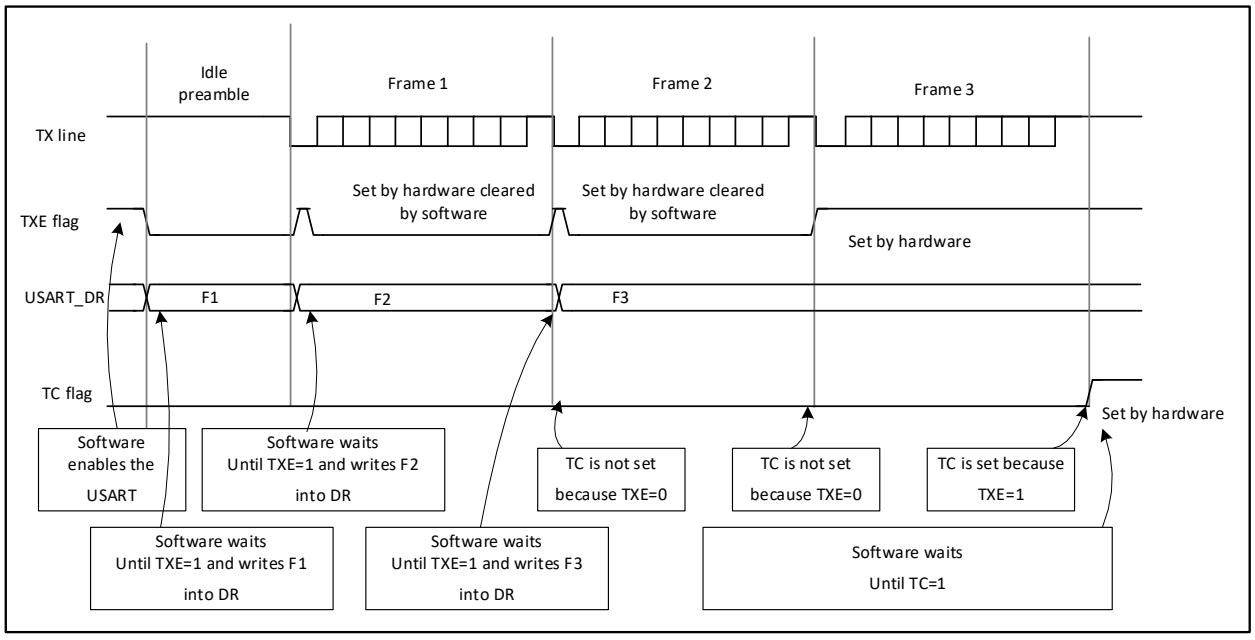

Figure 27-4 TC/TXE behavior when transmitting

27.3.2.4. Break characters

Setting the SBK bit transmits a break character. The break frame length depends on the M bits. If a ‘1’ is written to the SBK bit, a break character is sent on the TX line after completing the current character transmission. The SBK bit is set by the write operation and it is reset by hardware when the break character is completed (during the stop bits after the break character). The USART inserts a logic 1 signal (STOP) for the duration of at the end of the break frame to guarantee the recognition of the start bit of the next frame.

If software resets the SBK bit before starting to transmit the break frame, the break symbol will not be sent. If two consecutive break frames are to be sent, the SBK bit should be set after the stop bit of the previous break symbol.

27.3.2.5. Idle characters

Setting the TE bit drives the USART to send an idle frame before the first data frame.

27.3.3. Recevier

The USART can receive data words of either 7, 8 or 9 bits depending on the M bits in the USART_CR1 register.

27.3.3.1. Start bit detection

In the USART, the start bit is detected when a specific sequence of samples is recognized. This sequence is: 1 1 1 0 X 0 X 0 X 0 0 0 0.

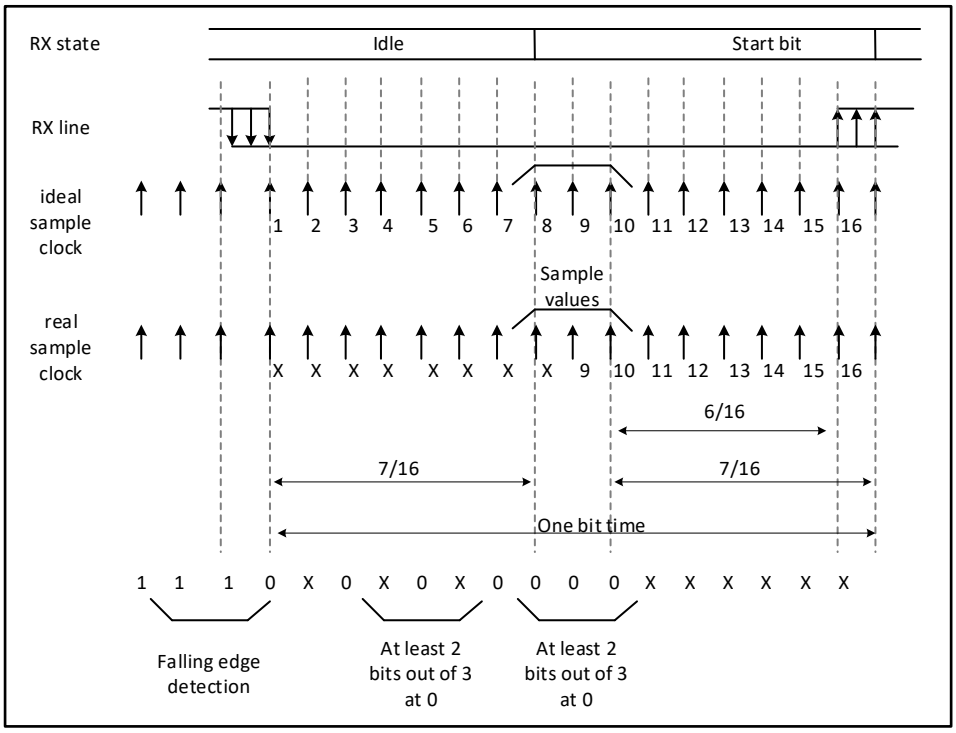

Figure 27-5 Start bit detection

If the sequence is incomplete, the receiver will exit the start bit detection and return to the idle state *(* without setting the flag *)* to wait for a falling edge. If all *3* sample points are *'0' (* the first sample at bits *3, 5,* and *7, and the second sample at bits 8*, *9*, and *10* are all *'0')*, then acknowledge receipt Start bit, then set *the RXNE* flag bit, if *RXNEIE = 1*, an interrupt will be generated.

If only *2 of the 3 sample points* are *'0' twice (* the *3rd, 5th, 7th* sample point and the *8th, 9th, 10th* sample point *),* then the start bit is still valid, but The *NE* noise flag is set. If this condition cannot be met, the detection process of the start bit is aborted, and the receiver will return to the idle state *(* the flag bit is not set *).*

If at one time only *2 of the 3 sample points* are *'0' (* the *3rd, 5th, 7th* sample point or the *8th, 9th, 10th* sample point *),* then the start bit is still valid, but the *NE* noise flag is set.

27.3.3.2. Character reception

During USART reception, the least significant bit of data is shifted in first from the RX pin. In this mode, the USART_DR register contains a buffer between the internal bus and the receive shift register.

Configuration steps:

1. Set UE in USART_CR1 register to 1 to activate USART.
2. Program the M bits of USART_CR1 to define the word length
3. Write the number of stop bits in USART_CR2
4. If multi-buffer communication is required, select the DMA enable bit (DMAR) in USART_CR3. Configure the DMA registers as required for multi-buffer communication.
5. Select the desired baud rate using the baud rate register USART_BRR.
6. Set the RE bit of USART_CR1. Activate the receiver to start looking for the start bit.

When a character is received :

● The RXNE bit is set. It indicates that the contents of the shift register are transferred to the RDR. In other words, the data has been received and can be read (including error flags associated with it).

● If the RXNEIE bit is set, an interrupt is generated.

● If a frame error, noise or overflow error is detected during reception, the error flag will be set

● In multi-buffer communication, RXNE is set up after each byte is received, and is cleared by the DMA read operation of the data register.

● In single buffer mode, the RXNE bit is cleared by software reading the USART_DR register. The RXNE flag can also be cleared by writing 0 to it. The RXNE bit must be cleared before the end of the next character reception to avoid overrun errors. Note: The *RE* bit should not be reset while receiving data. If the *RE* bit is cleared on reception, the reception of the current byte is lost.

27.3.3.3. Break character

When a break character is received, the USART handles it as a framing error.

27.3.3.4. Idle character

When an idle frame is detected, there is the same procedure as for a received data character plus an interrupt if the IDLEIE bit is set.

27.3.3.5. Overrun error

An overrun error occurs when a character is received when RXNE has not been reset. Data can not be transferred from the shift register to the RDR register until the RXNE bit is cleared. The RXNE flag is set after every byte received. An overrun error occurs if RXNE flag is set when the next data is received or the previous DMA request has not been serviced.

When an overrun error occurs:

● The ORE bit is set.

● The RDR content will not be lost. The previous data is available when a read to USART_RDR is performed.

● The shift register will be oveRWritten. After that point, any data received during overrun is lost.

● An interrupt is generated if either the RXNEIE bit is set or both the EIE and DMAR bits are set.

● Sequential read operations of USART_SR and USART_DR registers can reset the ORE bit

Note: When the *ORE* bit is set, it indicates that at least *1* data has been lost. There are two possibilities:

● If RXNE = 1, the last valid data is still in the receive register RDR and can be read.

● If RXNE = 0, it means that the last valid data has been read, and there is nothing to read in RDR. This can happen when new (ie lost) data is received while the last valid data is being read in the RDR. This can also happen when new data is received during the read sequence (between the USART_SR register read access and the USART_DR read access).

27.3.3.6. Noise error

Data recovery is performed by distinguishing between valid input data and noise using oversampling techniques (except synchronous mode).

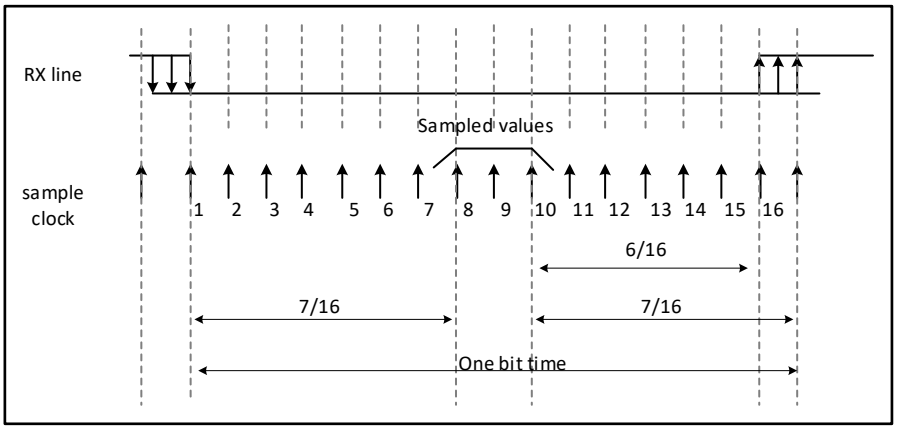

Figure 27-6 Data sampling for noise detection

Table 27-1 Noise detection from sampled data

Sample value	NE state	Bit value received	Data validity
000	0	0	Valid
001	1	0	Not Valid
010	1	0	Not Valid
011	1	1	Not Valid
100	1	0	Not Valid
101	1	1	Not Valid
110	1	1	Not Valid
111	0	1	Valid

When noise is detected in the received frame:

● Set the NE flag on the rising edge of the RXNE bit.

● Invalid data is transferred from the shift register to the USART_DR register.

● In the case of single-byte communication, no interrupt is generated. However, since the NE flag and the RXNE flag are set at the same time, RXNE will generate an interrupt. In the case of multi-buffer communication, an interrupt will be generated if the EIE bit in the USART_CR3 register has been set.

First read USART_SR, then read USART_DR register, will clear the NE flag bit.

27.3.3.7. Framing error

A framing error is detected when the stop bit is not recognized on reception at the expected time, following either a de-synchronization or excessive noise.

When the framing error is detected:

● The FE bit is set by hardware

● The invalid data is transferred from the Shift register to the USART_RDR register.

● No interrupt is generated in case of single byte communication. However this bit rises at the same time as the RXNE bit which itself generates an interrupt. In case of multibuffer communication an interrupt will be issued if the EIE bit is set in the USART_CR3 register.

Sequential reads of the USART_SR and USART_DR registers reset the FE bit.

27.3.3.8. Configurable stop bits during reception

The number of stop bits to be received can be configured through the control bits of Control Register 2 - it can be either 1 or 2 in normal mode.

- 1 stop bit: Sampling for 1 stop Bit is done on the 8th, 9th and 10th samples.
- 2 stop bits: Sampling for 2 stop bits is done on the 8th, 9th and 10th samples of the first stop bit. If a framing error is detected during the first stop bit the framing error flag will be set. The second stop bit is not checked for framing error. The RXNE flag will be set at the end of the first stop bit.

27.3.4. USART baud rate generation

The baud rate for the receiver and transmitter (Rx and Tx) are both set to the same value as programmed in the USART_BRR register.

$$\text{Tx / Rx baud} = fCK / (16 * USARTDIV)$$

Here *fCK* is the clock to the peripheral USARTDIV is an unsigned fixed-point number. The 12-bit value is set in the USART_BRR register.

Note: After writing to *USART_BRR,* the baud rate counter is replaced by the new value of the baud rate register. Therefore, do not change the value of the baud rate register while communication is in progress.

How to derive USARTDIV from USART_BRR register values

Example 1:

If DIV_Mantissa = 27, DIV_Fraction = 12 (USART_BRR = 0x1BC), then:

Mantissa (USARTDIV) = 27

Fraction (USARTDIV) = 12/16 = 0.75

Therefore USARTDIV = 27.75

Example 2:

To program USARTDIV = 25.62, then:

DIV_Fraction = 16*0.62 = 9.92

The nearest integer is: 10 = 0x0A

DIV_Mantissa = mantissa (25.620) = 25 = 0x19

Then, USART_BRR = 0x19A

Example 3:

To program USARTDIV = 50.99, then:

DIV_Fraction = 16*0.99 = 15.84

The nearest integer is: 16 = 0x10 = > DIV_frac[3:0] overflow = > carry must be added to the fractional part

DIV_Mantissa = mantissa (50.990 + carry) = 51 = 0x33

Then, USART_BRR = 0x330, USARTDIV = 51

Baud rate		F_{PCLK} = 36 MHz			F_{PCLK} = 72 MHz		
S.No	Kbps	Actual	BRR	Error(%)	Actual	BRR	Error(%)
1	2.4	2.400	937.5	0%	2.4	1875	0%
2	9.6	9.600	234.375	0%	9.6	468.75	0%

3	19.2	19.2	117.1875	0%	19.2	234.375	0%
4	57.6	57.6	39.0625	0%	57.6	78.125	0%
5	115.2	115.384	19.5	0.15%	115.2	39.0625	0%
6	230.4	230.769	9.75	0.16%	230.769	19.5	0.16%
7	460.8	461.538	4.875	0.16%	461.538	9.75	0.16%
8	921.6	923.076	2.4375	0.16%	923.076	4.875	0.16%
9	2250	2250	1	0%	2250	2	0%
10	4500	N.A	N.A	N.A	4500	1	0%

Note: The lower the CPU clock the lower the accuracy for a particular baud rate. The upper limit of the achievable baud rate can be fixed with these data.

27.3.5. USART receiver's tolerance to clock deviation

The asynchronous receiver of the USART works correctly only if the total clock system deviation is less than the tolerance of the USART receiver. The causes which contribute to the total deviation are:

- DTRA: Deviation due to the transmitter error (which also includes the deviation of the transmitter's local oscillator)
- DQUANT: Error due to the baud rate quantization of the receiver
- DREC: Deviation of the receiver's local oscillator
- DTCL: Deviation due to the transmission line (generally due to the transceivers which can introduce an asymmetry between the low-to-high transition timing and the high-tolow transition timing)

DTRA + DQUANT + DREC + DTCL < USART receiver's tolerance.

For normal reception of data, the tolerance of the USART receiver is equal to the maximum tolerable variation, which depends on the following choices:

- 10- or 11-bit character length defined by the M bits of the USART_CR1 register
- whether to use fractional baud rate to generate

Table 27-2 USART receiver tolerance when DIV_Fraction is 0

M bit	NF is an error	NF is don't care
0	3.75%	4.375%
1	3.41%	3.97%

Table 27-3 USART receiver tolerance when DIV_Franction is different from 0

M bit	NF is an error	NF is don't care
0	3.33%	3.88%
1	3.03%	3.53%

27.3.6. USART auto baud rate detection

The USART is able to detect and automatically set the USART_BRR register value based on the reception of one character. Automatic baud rate detection is useful under two circumstances:

1)The communication speed of the system is not known in advance

2)The system is using a relatively low accuracy clock source and this mechanism allows the correct baud rate to be obtained without measuring the clock deviation.

The clock source frequency must be compatible with the expected communication speed (oversampling by 16 must be selected and baudrate between fCK/65535 and fCK/16.

Before activating the auto baud rate detection, the auto baud rate detection mode must be chosen. There are various modes based on different character patterns(They can be chosen through the ABRMOD[1:0] field in the USART_CR3 register). In these auto baud rate modes, the baud rate is measured several times during the synchronization data reception and each measurement is compared to the previous one.

These modes are:

Mode 0: Any character starting with a bit at 1. In this case the USART measures the duration of the Start bit (falling edge to rising edge).

Mode 1: Any character starting with a 10xx bit pattern. In this case, the USART measures the duration of the Start and of the 1st data bit. The measurement is done falling edge to falling edge, ensuring better accuracy in the case of slow signal slopes.

In parallel, another check is performed for each intermediate transition of RX line. An error is generated if the transitions on RX are not sufficiently synchronized with the receiver (the receiver being based on the baud rate calculated on bit 0).

Prior to activating auto baud rate detection, the USART_BRR register must be initialized by writing a non-zero baud rate value.

The automatic baud rate detection is activated by setting the ABREN bit in the USART_CR2 register. The USART will then wait for the first character on the RX line. The auto baud rate operation completion is indicated by the setting of the ABRF flag in the USART_ISR register. If the line is noisy, the correct baud rate detection cannot be guaranteed. In this case the BRR value may be corrupted and the ABRE error flag will be set. This also happens if the communication speed is not compatible with the automatic baud rate detection range (bit duration not between 16 and 65536 clock periods (oversampling by 16) and not between 8 and 65536 clock periods (oversampling by 8)).

The RXNE interrupt will signal the end of the operation. At any later time, the auto baud rate detection may be relaunched by resetting the ABRF flag (by writing a 0).

Note: If the USART is disabled (UE = 0) during an auto baud rate operation, the BRR value may be corrupted.

27.3.7. Multiprocessor communication using USART

It is possible to perform multiprocessor communication with the USART (with several USARTs connected in a network). For instance one of the USARTs can be the master, its TX output connected to the RX inputs of the other USARTs. The others are slaves, their respective TX outputs are logically ANDed together and connected to the RX input of the master.

In multiprocessor configurations it is often desirable that only the intended message recipient should actively receive the full message contents, thus reducing redundant USART service overhead for all non addressed receivers.

The non addressed devices may be placed in mute mode by means of the muting function.In mute mode:

- None of the reception status bits can be set.
- All the receive interrupts are inhibited.
- The RWU bit in USART_ISR register is set to 1. RWU can be controlled automatically by hardware or by software, through the MMRQ bit in the USART_RQR register, under certain conditions. The USART can enter or exit from mute mode using one of two methods, depending on the WAKE bit in the USART_CR1 register:
- Idle Line detection if the WAKE bit is reset.
- Address Mark detection if the WAKE bit is set.

27.3.7.1. Idle line detection (WAKE = 0)

The USART enters mute mode when the RWU bit is written to 1. It wakes up when an Idle frame is detected. Then the RWU bit is cleared by hardware but the IDLE bit is not set in the USART_ISR register. An example of mute mode behavior using Idle line detection is given in Figure 27-7.

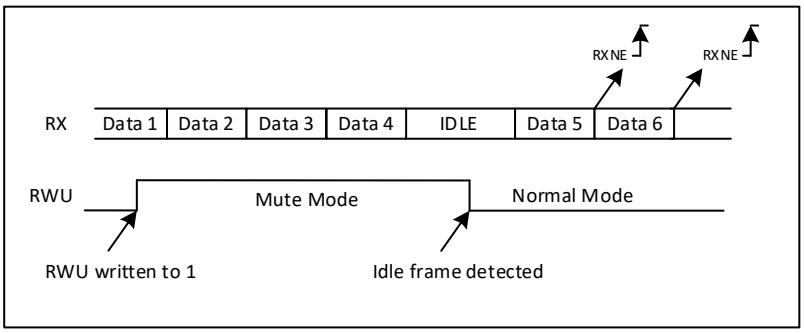

Figure 27-7 Mute mode using Idle line detection

27.3.7.2. Address mark check (WAKE = 1)

In this mode, bytes are recognized as addresses if their MSB is a ‘1’ otheRWise they are considered as data. In an address byte, the address of the targeted receiver is put in the 4 LSBs. The choice of 4-bit address detection is done using the ADDM7 bit. This 4-bit/7-bit word is compared by the receiver with its own address which is programmed in the ADD bits in the USART_CR2 register.

If the received byte does not match its programmed address, the USART enters silent mode. At this point, the hardware sets the RWU bit.

Receiving this byte will neither set the RXNE flag nor generate an interrupt or issue a DMA request because the USART is already in silent mode.

When the received byte matches the programmed address in the receiver, the USART exits silent mode. Then the RWU bit is cleared and subsequent bytes are received normally. The RXNE bit will be set when this matching address byte is received because the RWU bit has been cleared.

When the receive buffer contains no data (RXNE = 0 in USART_SR), the RWU bit can be written to 0 or 1. OtheRWise, the write operation is ignored. The figure below shows an example of using address mark detection to wake up and enter silent mode.

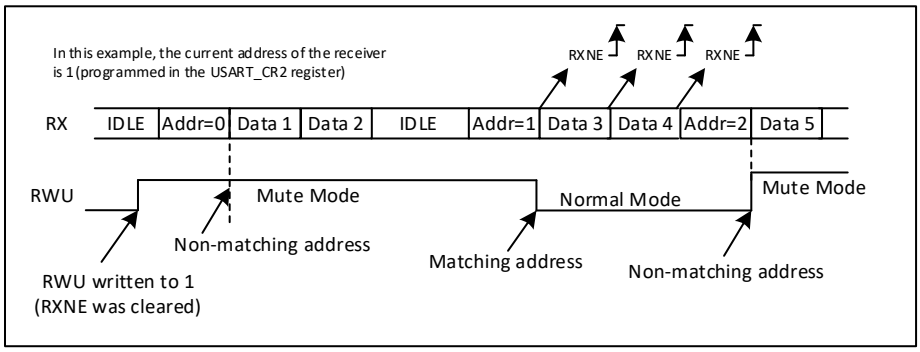

Figure 27-8 Slient mode with address tag detection

27.3.7.3. Check control

Setting the PCE bit on the USART_CR1 register enables parity control (generates a parity bit when transmitting, and performs parity checking when receiving). The possible USART frame formats are listed in the table below according to the frame length defined by the M bits.

Table 27-4 Frame format

M bit	PCE bit	USART fram
0	0	SB—8 bit data—STB
0	1	SB—7 bit data—PB—STB
1	0	SB—9 bit data—STB
1	1	SB—8 bit data—PB—STB

When waking up a device with an address tag, the address is matched only considering the *MSB* bits of the data, not the parity bits. *(MSB* is the last sent out of the data bits, followed by the parity bit or stop bit*)*

27.3.7.4. Even parity

The parity bit is calculated to obtain an even number of “1s” inside the frame of the 7 or 8 LSB bits (depending on M bits values) and the parity bit.

As an example, if data = 00110101, and 4 bits are set, then the parity bit will be 0 if even parity is selected (PS bit in USART_CR1 = 0).

27.3.7.5. Odd parity

The parity bit is calculated to obtain an odd number of “1s” inside the frame made of the 7 or 8 LSB bits (depending on M bits values) and the parity bit.

As an example, if data = 00110101 and 4 bits set, then the parity bit will be 1 if odd parity is selected (PS bit in USART_CR1 = 1).

27.3.7.6. Transfer mode

If the PCE bit is set in USART_CR1, then the MSB bit of the data written in the data register is transmitted but is changed by the parity bit (even number of “1s” if even parity is selected or an odd number of “1s” if odd parity is selected). If the parity check fails, the PE flag is set in the USART_ISR register and an interrupt is generated if PEIE is set in the USART_CR1 register. The PE flag is cleared by software riting 1 to the PECF in the USART_ICR register.

27.3.8. USART synchronous mode

The synchronous mode is selected by writing the CLKEN bit in the USART_CR2 register to 1. In synchronous mode, the following bits must be kept cleared:

● The HDSEL bit in USART_CR3 register

The USART allows the user to control bidirectional synchronous serial communications in a master mode. The CK pin is the output of the USART transmitter clock. During the start and stop bits, there is no clock pulse on the CK pin. Depending on the state of the LBCL bit in the USART_CR2 register, a clock pulse is generated or not generated during the last valid data bit. The CPOL bit in the USART_CR2 register allows the user to select the clock polarity, and the CPHA bit in the USART_CR2 register allows the user to select the phase of the external clock.

The external CK clock is not activated during bus idle periods, before actual data arrives and when the disconnect symbol is sent.

In synchronous mode, the USART transmitter works exactly the same as in asynchronous mode. But because CK is synchronous with TX (according to CPOL and CPHA), data on TX is sent synchronously with CK.

The USART receiver in synchronous mode works differently than in asynchronous mode. If RE = 1, the data is sampled on CK (rising or falling according to CPOL and CPHA) without any oversampling. But setup time and duration (depending on baud rate, 1/16 bit time) must be considered.

Notice:

CK pin works together with the *TX* pin. Thus, the clock is only provided when the transmitter is enabled *(TE = 1)* and data is being sent *(* writing data to *the USART_DR* register *).* This means that it is impossible to receive a sync data without sending data.

LBCL, CPOL and *CPHA* bits should be when both the transmitter and receiver are disabled, these bits cannot be changed when the transmitter or receiver is enabled.

TE and *RE* in the same instruction to reduce receiver setup and hold time.

USART only supports master mode: it cannot receive or transmit data with an input clock from other devices *(CK* is always an output*).*

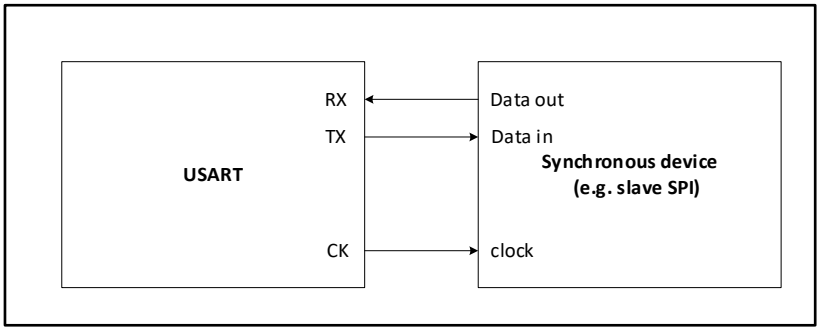

Figure 27-9 Example of USART isochronous transmission

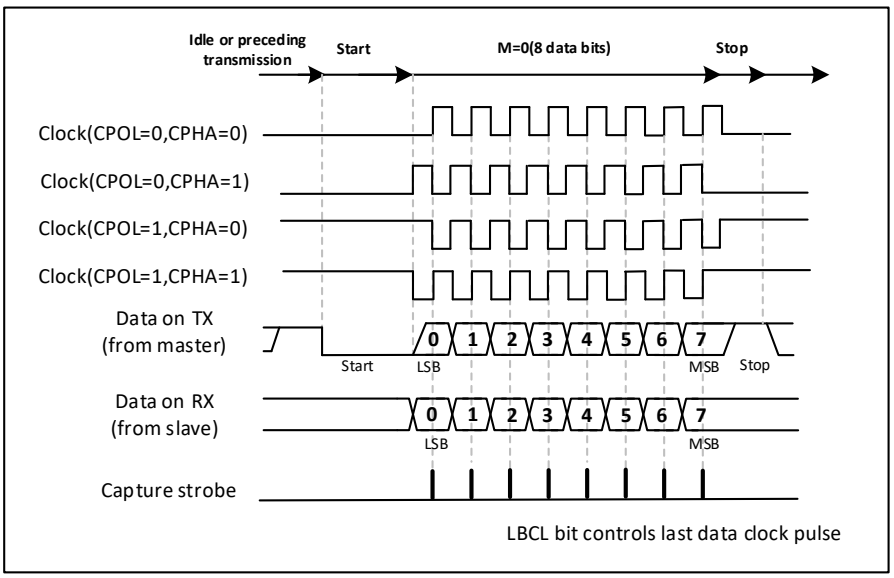

Figure 27-10 USART data clock timing example (M = 0)

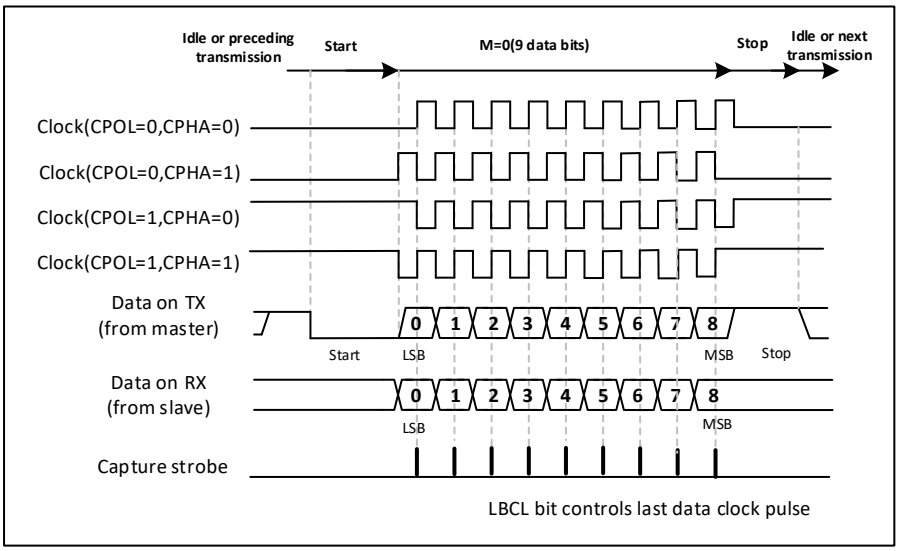

Figure 27-11 USART data clock timing example(M = 1)

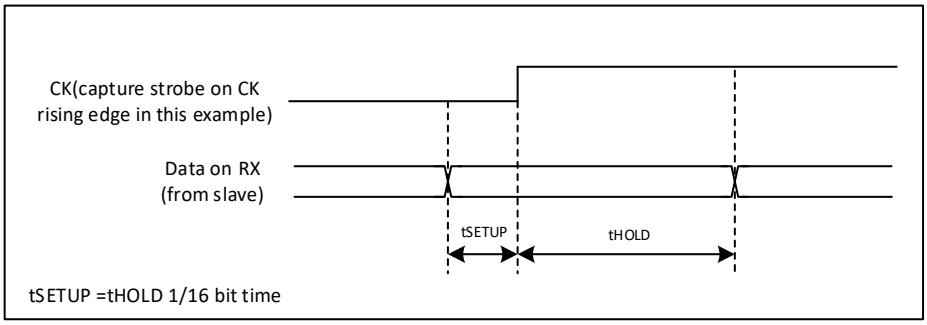

Figure 27-12 RX data sample/hold time

27.3.9. USART single-wire half-duplex communication

Single-wire Half-duplex mode is selected by setting the HDSEL bit in the USART_CR3 register. In this mode, the following bits must be kept cleared:

- LINEN and CLKEN bits in the USART_CR2 register

The USART can be configured to follow a Single-wire Half-duplex protocol where the TX and RX lines are internally connected. The selection between half- and Full-duplex communication is made with a control bit HDSEL in USART_CR3.

As soon as HDSEL is written to 1:

- The RX pin is no longer used
- The TX pin is always released when no data is transmitted. Thus, it acts as a standard I/O in idle or in reception. It means that the I/O must be configured so that TX is configured as alternate function open-drain with an external pull-up.

Apart from this, the communication protocol is similar to normal USART mode. Any conflicts on the line must be managed by software (by the use of a centralized arbiter, for instance). In particular, the transmission is never blocked by hardware and continues as soon as data is written in the data register while the TE bit is set.

27.3.10. USART continuous communication in DMA mode

The USART is capable of performing continuous communication using the DMA. The DMA requests for Rx buffer and Tx buffer are generated independently.

27.3.10.1. Transmission using DMA

DMA mode can be enabled for transmission by setting DMAT bit in the USART_CR3 register. Data is loaded from a SRAM area configured using the DMA peripheral to the USART_DR register whenever the TXE bit is set. To map a DMA channel for USART transmission, use the following procedure (x denotes the channel number):

1) Write the USART_TDR register address in the DMA control register to configure it as the destination of the transfer. The data is moved to this address from memory after each TXE event.

2) Write the memory address in the DMA control register to configure it as the source of the transfer. The data is loaded into the USART_TDR register from this memory area after each TXE event.

3) Configure the total number of bytes to be transferred to the DMA control register.

4) Configure the channel priority in the DMA register.

5) Configure DMA interrupt generation after half/ full transfer as required by the application.

6) Clear the TC flag in the USART_ISR register by setting the TCCF bit in the USART_ICR register.

7) Activate the channel in the DMA register.

When the number of data transfers programmed in the DMA Controller is reached, the DMA controller generates an interrupt on the DMA channel interrupt vector.

In transmission mode, once the DMA has written all the data to be transmitted, the TC flag can be monitored to make sure that the USART communication is complete. This is required to avoid corrupting the last transmission before disabling the USART or entering Stop mode. software needs to wait for TXE = 1 first, then wait for TC = 1.

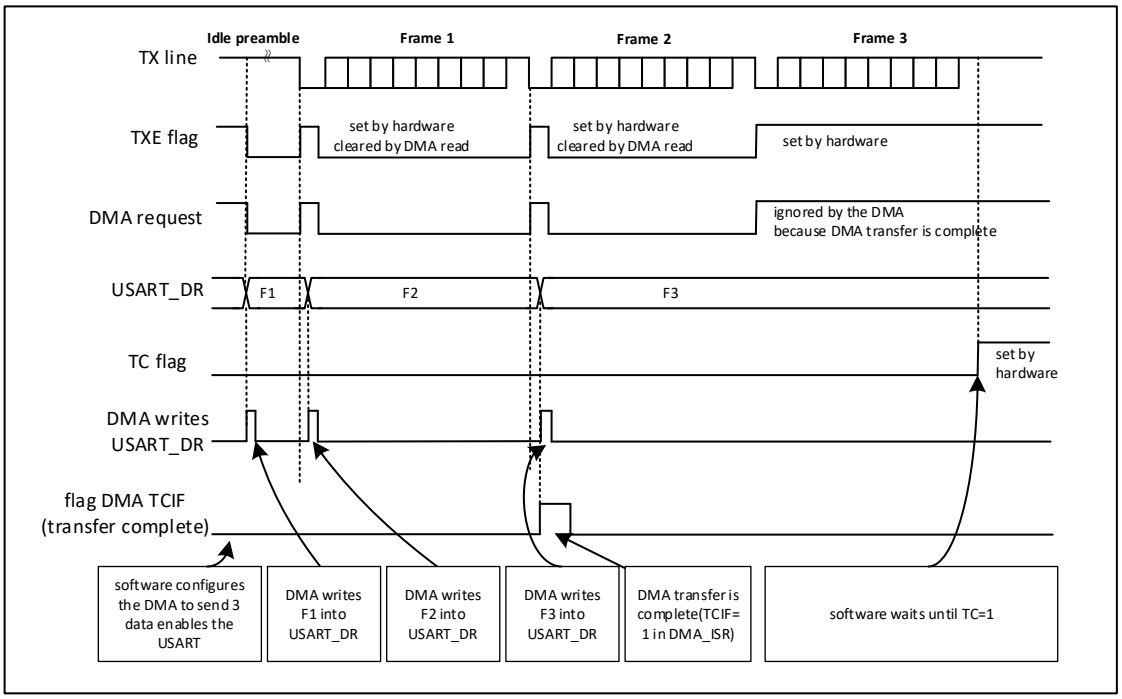

Figure 27-13 Send using DMA

27.3.10.2. Reception using DMA

DMA can be activated by setting the DMAR bit of the USART_CR3 register for receiving. Each time a byte is received, the DMA controller will transfer the data from the USART_DR register to the specified SRAM area (refer to DMA related instructions). The steps to assign a DMA channel for USART reception are as follows (x represents the channel number):

1) Configure the USART_DR register address as the source address of the transfer through the DMA control register. After each RXNE event, data will be read from this address and transferred to memory.

2) Configure the memory address as the destination address of the transfer through the DMA control register. Data will be transferred from USART_DR to this memory area after each RXNE event.

3) Configure the total number of bytes to be transferred in the DMA control register.

4) Configure the channel priority on the DMA register.

5) According to the requirements of the application, configure the DMA interrupt to be generated when the transfer is half or fully completed.

6) Activate the channel on the DMA control register.

When receiving the transfer amount specified by the DMA controller, the DMA controller generates an interrupt on the interrupt vector of the DMA channel.

27.3.10.3. Error flags and interrupt generation in multi-buffer communication

In the case of multi-buffer communication, if any error occurs during the communication, the error flag will be set after the current byte has been transferred. An interrupt will be generated if the interrupt enable bit is set. In the case of a single byte reception, the framing error, overflow error and noise flags that are set together with RXNE have separate error flag interrupt enable bits, if set, an interrupt will be generated after the current byte transmission is completed.

27.3.11. Hardware flow control

Using the nCTS input and nRTS output. The figure below shows how to connect 2 devices in this mode.

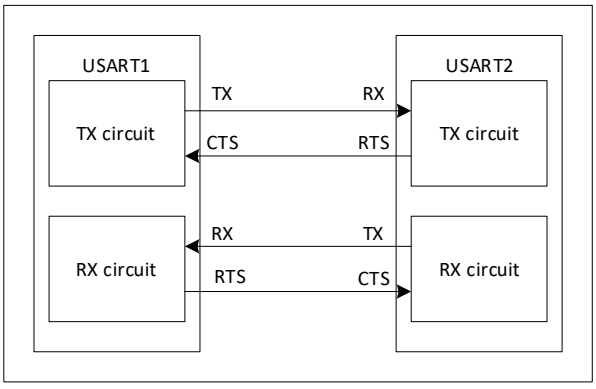

Figure 27-14 Hardware flow control between two USARTs

27.3.11.1. RTS flow control

If RTS flow control is enabled (RTSE = 1), nRTS becomes active (connected low) as soon as the USART receiver is ready to receive new data. When data arrives in the receive register, nRTS is released, thereby indicating that data transmission is to be stopped at the end of the current frame.

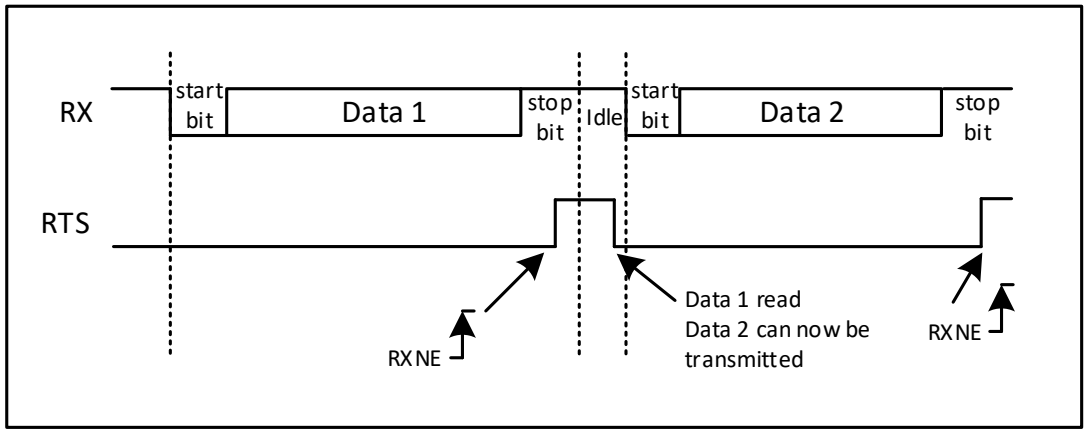

Figure 27-15 RTS flow control

27.3.11.2. CTS flow control

If CTS flow control is enabled (CTSE = 1), the transmitter checks the nCTS input before sending the next frame. If nCTS is valid (pulled to a low level), the next data is sent (assuming that data is ready to be sent, that is, TXE = 0), otheRWise the next frame of data is not sent. If the nCTS is invalidated during transmission, the transmission stops after the current transmission is completed.

When CTSE = 1, as long as the nCTS input changes state, the hardware automatically sets the CTSIF status bit. It indicates whether the receiver is ready to communicate. An interrupt is generated if the CTSIE bit in the USART_CT3 register is set.

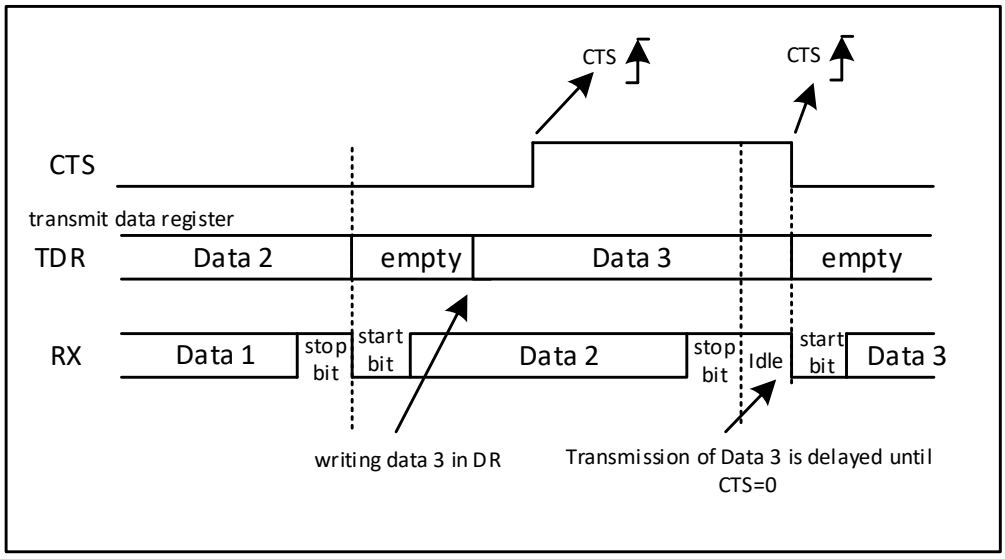

Figure 27-16 CTS flow control

27.4. USART interrupt request

Serial number	Interrupt event	Event flag	Enable bit	Send/receive
1	Send data register empty	TXE	TXEIE	Send
2	CTS (Clear to Send)interrupt	CTSIF	CTSIE	Send

3	Transmission completed	TC	TCIE	Send
4	The receive register is not empty (read data is ready)	RXNE	RXNEIE	Take over
5	Overrun error	ORE		Take over
6	Idle frame	IDLE	IDLEIE	Take over
7	Parity error	PE	PEIE	Take over
8	Noise, overrun and frame errors when communicating with multiple processors	NR/ORE/FE	EIE	Take over

All USART interrupts share the same interrupt vector.

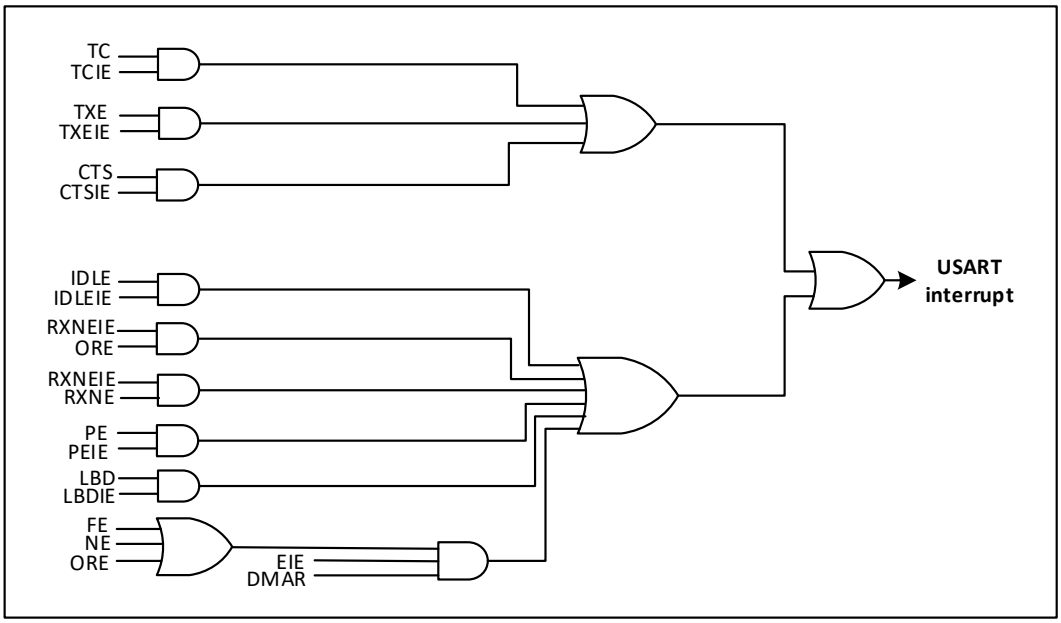

Figure 27-17 USART interrupt map

27.5. USART register

27.5.1. Status register (USART_SR)

Address offset: 0x00

Reset value: 0x0000 00C0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	ABRRQ	ABRE	ABRF	CTS	Res	TXE	TC	RXNE	IDLE	ORE	NE	FE	PE
			W	R	R	RC_W0		R	RC_W0	RC_W0	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	RES	-	Reserved
12	ABRRQ	W	0	Automatic baud rate request Writing 1 to this bit resets the ABRF flag and requests automatic baud rate detection for the next frame.
11	ABRE	R	0	Autobaud error flag. This register is set by hardware when there is an error in automatic baud rate detection (baud rate out of range or character comparison error). Software clears this bit by writing a 1 to the ABRRQ register.
10	ABRF	R	0	Automatic baud rate detection flag.

				This bit is set to 1 by hardware when auto-baud rate is set (set RXNE = 1 at the same time, an interrupt will be generated when the interrupt is enabled), or when an error occurs in the auto-baud rate detection operation (ABRE = 1, RXNE = 1, FE = 1). Software clears this bit by writing a 1 to the ABRRQ bit in the USART_RQR register.
9	CTS	RC_W0	0	When CTS input toggle, do not CTSE = 1, this register is 1. Software writes 0 to clear. When CTSIE = 1, a CTS interrupt is generated. 0: CTS line value unchanged 1: CTS line value change
8	reserved			
7	TXE	R	1	Transfer register empty flag. This register is set by hardware when the USART_DR register data is transferred to the shift register. When TXEIE = 1, an interrupt is generated. Writing to the USART_DR register will clear this bit 0: Data is not transferred to the shift register 1: Data is transferred to the shift register
6	TC	RC_W0	1	Transmission complete flag. After the transmission of the data frame is completed, and TXE = 1, the hardware will set this register. An interrupt is generated when TCIE = 1. Software reading the USART_SR register first and then writing the USART_DR register will clear this bit (for multiprocessor communication). Software can also write 0 to clear. 0: Transmission not completed 1: Transmission completed
5	RXNE	RC_W0	0	The read data register is not empty flag. This register is set by hardware when the shift register value is transferred to the USART_DR register. Software reads the USART_DR register, or writes 0 to clear this bit. When RXNEIE = 1, an interrupt is generated. 0: No data received 1: Receive data ready
4	IDLE	R	0	Idle sign. Detect IDLE line, the hardware sets this register. An interrupt is generated when IDLEIE = 1. Software can clear this bit by reading the USART_SR register first and then the USART_DR register. 0: IDLE not detected line 1: IDLE detected line
3	ORE	R	0	Overrun error flag. When RXNE = 1, the hardware sets this bit when the data received in the shift register is about to be transferred to the RDR register. Software can clear this bit by reading the USART_SR register first and then the USART_DR register. When RXNEIE = 1, an interrupt is generated. 0: No Overrun error is generated 1: Generate Overrun error Note: When this register is set, the contents of the RDR register are not lost, but the contents of the shift register are oveRWritten. When EIE = 1, an ORE interrupt is generated.
2	NE	R	0	Noise error sign. This register is set by hardware when the data frame receives noise. Software can clear this bit by reading the USART_SR register first and then the USART_DR register. 0: No noise error detected

				1: Noise error detected Note: When RXNE and NE are generated at the same time, no interrupt is generated when NE = 1, but an interrupt is generated when the RXNE flag is set. In multi-buffer communication mode, NE = 1 will generate an interrupt when EIE = 1.
1	FE	R	0	Framing error flag. This bit is set by hardware when out-of-sync, excessive noise, or abort characters are detected. Software can clear this bit by reading the USART_SR register first and then the USART_DR register. 0: no frame error detected 1: Framing error or break character detected Note: When RXNE and FE are generated at the same time, no interrupt is generated when FE = 1, but an interrupt is generated when the RXNE flag is set. If the currently transmitted data has both a frame error and an overload error, the hardware will continue to transmit the data and only set the ORE flag. In multi-buffer communication mode, FE = 1 will generate an interrupt when EIE = 1.
0	PE	R	0	Checksum error. This register is set by hardware when the parity value is incorrect during reception. Software can clear this bit by reading the USART_SR register first and then the USART_DR register. But software must wait for RXNE = 1 before clearing this bit. When PEIE, an interrupt is generated. 0: No parity error is generated 1: Generate parity error

27.5.2. UASRT data register (USART_DR)

Address offset: 0x04

Reset value: undefined

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	DR[8:0]								
							RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:9	Reserved	RES	-	Reserved
8:0	DR[8:0]	RW	undefined	Receive/transmit data register. Depending on the read or write operation, the former is the received data and the latter is the transmitted data. The DR register is physically composed of two registers (one is the transmitted T DR, the other is the received R DR), so the DR register implements two functions of reading and writing. T DR register provides a parallel interface between the internal bus and the output shift register, and the R DR register provides a parallel interface between the input shift register and the internal bus. When parity is enabled for transmit operation, writing the M SB bit (bit7 or bit8) has no effect because it has been replaced by the parity bit. When the parity enable is turned on for a receive operation, the read MSB bit is the received parity bit.

27.5.3. Baud rate register (USART_BRR)

Address offset: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DIV_Mantissa[11:0]												DIV_Faction[3:0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

In auto-baud detection mode, hardware updates this register.

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	RES	-	Reserved
15:4	DIV_Mantissa[15:4]	RW	0	12bit integer
3:0	DIV_Fraction[3:0]	RW	0	4bit decimal

27.5.4. USART control register 1 (USART_CR1)

Address offset: 0x0C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	UE	M	WAKE	PCE	PS	PEIE	TXEIE	TCIE	RXNEIE	IDLEIE	TE	RE	RWU	SBK
		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:14	Reserved	RES	-	Reserved
13	UE	RW	0	USART enabled. When this bit is cleared, the USART module will immediately stop the current operation. This bit is set and cleared by software. 0: USART prescaler and output disabled, low-power mode 1: USART enable The software needs to wait for USART_ISR.TC to be set before clearing the UE bit and entering the low power mode, Also, the DMA channel needs to be disabled before clearing the UE bit.
12	M	RW	0	0: 1 start bit, 8 data bits, n stop bit 1: 1 start bit, 9 data bit, n stop bit
11	WAKE	RW	0	Receive wakeup mode. How to wake up from mute mode. Set or cleared by software. 0: Idle line wake up 1: address wake-up
10	PCE	RW	0	Parity control. 0: Parity check disabled 1: Parity check enabled Parity bit: 9th bit of 9bit, 8th bit of 8bit.
9	PS	RW	0	Parity check selection. Set and cleared by software. 0: Even parity 1: odd parity
8	PEIE	RW	0	PE interrupt enable. Set and cleared by software. 0: Disable 1: PE interrupt enable
7	TXEIE	RW	0	TXE interrupt enable. Set and cleared by software. 0: Disable 1: TX E interrupt enable

6	TCIE	RW	0	End of transfer interrupt enable. Set and cleared by software. 0: Disable 1: TC interrupt enable
5	RXNEIE	RW	0	RXNE interrupt enable, set and cleared by software. 0: Disable 1: ORE or RXNE interrupt enable
4	IDLEIE	RW	0	IDLE interrupt enable. Set and cleared by software. 0: Disable 1: IDLE interrupt enable
3	TE	RW	0	Transmission enable. 0: Transmission prohibited 1: Transmission enable
2	RE	RW	0	Receive enable. 0: Reception prohibited 1:Receive enable, start to detect start bit
1	RWU	RW	0	Receive wakeup. This bit indicates whether the USART is in mute mode. This register is set when a mute mode sequence is received, this register is cleared when a wake-up sequence is received. The specific wake-up sequence (address or IDLE) is controlled by the register USART_CR1.WAKEbit. 0: The receiver is in working mode 1: The receiver is in silent mode Note 1: Before setting this bit to enter mute mode, the USART must have received a data byte, otheRWise in mute mode, it cannot be woken up by idle bus detection. Note 2: When configured as address mark detection wake-up (WAKE = 1), the RWU bit cannot be modified by software when RXNE is set.
0	SBK	RW	0	Send break frame. Software sets this register to send the break byte. This register is cleared by hardware after the stop bit of the break frame is sent. 0: do not send break bytes 1: send break byte

27.5.5. USART control register 2 (USART_CR2)

Address offset: 0x10

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	STOP[1:0]		CLKEN	CPOL	CPHA	LBCL	Res	Res	Res	Res	ADD[3:0]			
		RW	RW	RW	RW	RW	RW					RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:14	Reserved	RES	-	Reserved
13	STOP	RW	0	Stop bit configuration. 0: 1 stop bit 1: 2 stop bit
12	Reserved			
11	CLKEN	RW	0	CK pin enable. 0: disable 1: CK pin enable This bit is reserved when synchronous mode is not supported.
10	CPOL	RW	0	Clock polarity. Sync mode, CK pin output clock polarity.

				0: Outside the transmission window, CK pin is a stable low value, 1: Outside the transmission window, CK pin is a stable high value,
9	CPHA	RW	0	This bit is used to select the phase of the CK pin output clock in synchronous mode. It works with the CPOL bits to generate the desired clock/data relationship. 0: The first clock transfer is the first data capture edge, 1: The second clock transfer is the first data capture edge,
8	LBCL	RW	0	Whether the clock pulse of the last bit of data is in CK pin output. 0: The clock pulse of the last bit of data is not in CK pin output, 1: The clock pulse of the last bit of data is at CK pin output.
7:4	Reserved	RES	-	Reserved
3:0	ADD[3:0]	RW	4'b0	USART address. This register is used in the mute mode of the multiprocessor, and is used as the address when the 4- bit address wakes up.

27.5.6. USART control register 3 (USART_CR3)

Address offset: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	ABR-MOD[1:0]		ABREN	OVER8	CTSIE	CTSE	RTSE	DMAT	DMAR	Res	Res	HDSEL	Res	Res	EIE
	RW		RW	RW	RW	RW	RW	RW	RW			RW			RW

Bit	Name	R/W	Reset Value	Function
31:5	Reserved	RES	-	Reserved
14:13	ABRMOD[1:0]	RW	2'b0	Automatic baud rate detection mode. 00: measure the baud rate from the start bit 01: Falling edge to falling edge measurement 10: Reserved 11: Reserved When ABREN = 0 or UE = 0, this register is write-only.
12	ABREN	RW	0	Auto-baud rate enabled. 0: Disable 1: Auto baud rate enabled
11	OVER8	RW	0	Oversampling mode. 0: Oversampling by 16 1: Oversampling by 8 can only be written when U E = 0.
10	CTSIE	RW	0	CTS interrupt enable. 0: Forbidden, 1: CTSIF interrupt enable,
9	CTSE	RW	0	CTS enabled. 0: CTS hardware flow control is disabled, 1: CTS mode enabled. Data is only transmitted when the CTS input is 0. At this point, after the data is written into the data register, the transmission will not be started until the CTS is valid.
8	RTSE	RW	0	RTS enabled. 0: RTS hardware flow control is disabled, 1: The RTS output is enabled, and the next data is requested only when the receive buffer is not

				full. After the current data is sent, the sending operation is suspended. If data can be received, set RTS to valid (0).
7	DMAT	RW	0	Enable DMA when transferring. 0: Forbidden, 1: Enable DMA during transfer.
6	DMAR	RW	0	DMA is enabled when receiving. 0: Forbidden, 1: Enable DMA when receiving.
5:4	reserved			
3	HDSEL	RW	0	Half-duplex option. 0: Non-half duplex mode, 1: Half-duplex mode selection.
2:1				
0	EIE	RW	0	Error interrupt enable. 0: Forbidden, 1: Frame error FE, overrun error ORE, noise NF interrupt enable.

27.5.7. USART register map

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	USART_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ABRRQ	ABRE	ABRF	CTS	Res.	TXE	TC	RXNE	IDLE	ORE	NE	FE	PE
	Reset value																				0	0	0	0		1	1	0	0	0	0	0	0
0x04	USART_DR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DR[8:0]								
	Reset value																								0	0	0	0	0	0	0	0	0
0x08	USART_BRR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DIV_Mantissa[11:0]												DIV_Faction[3:0]			
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	USART_CR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	UE	M	WAKE	PCE	PS	PEIE	TXEIE	TCIE	RXNEIE	IDLEIE	TE	RE	RWU	SBK
	Reset value																			0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x10	USART_CR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	STOP	Res.	CLKEN	CPOL	CPHA	LBCL	Res.	Res.	Res.	Res.	ADD[3:0]			
	Reset value																			0		0	0	0	0					0	0	0	0
0x14	USART_CR3	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ABR-MOD[1:0]		ABREN	OVER8	CTSIE	CTSE	RTSE	DMAT	DMAR	Res.	Res.	HDSEL	Res.	Res.	EIE
	Reset																		0	0	0	0	0	0	0	0	0			0			0

	value																															
0x18	Re-served	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.

28. Serial peripheral interface (SPI)

This project designs and implements two SPI modules, both of which have exactly the same functions.

28.1. Introduction

Serial Peripheral Interface (SPI) allows the chip to communicate with external devices in half-duplex, full-duplex, and simplex synchronous serial communication. This interface can be configured in master mode and provides the communication clock (SCK) for external slave devices. The interface can also work in a multi-master configuration.

It can be used for a variety of purposes, including two-wire simplex simultaneous transmission using one bidirectional data line.

28.2. SPI main features

- Master or slave operation
- Full-duplex synchronous transfers on three lines
- Half-duplex synchronous transfer on two lines (with bidirectional data line)
- Simplex synchronous transfers on two lines (with unidirectional data line)
- 8-bit to 16-bit data size selection
- Multimaster mode capability
- 8 master mode baud rate prescalers up to fPCLK/4.
- Slave mode frequency up to fPCLK/4.
- NSS management by hardware or software for both master and slave: dynamic change of master/slave operations
- Programmable clock polarity and phase
- Programmable data order with MSB-first or LSB-first shifting
- Dedicated transmission and reception flags with interrupt capability
- SPI bus busy status flag
- SPI Motorola support
- Master mode fault, overrun flags with interrupt capability
- Two 32-bit embedded Rx and Tx FIFOs with DMA capability

28.3. SPI function description

28.3.1. Overview

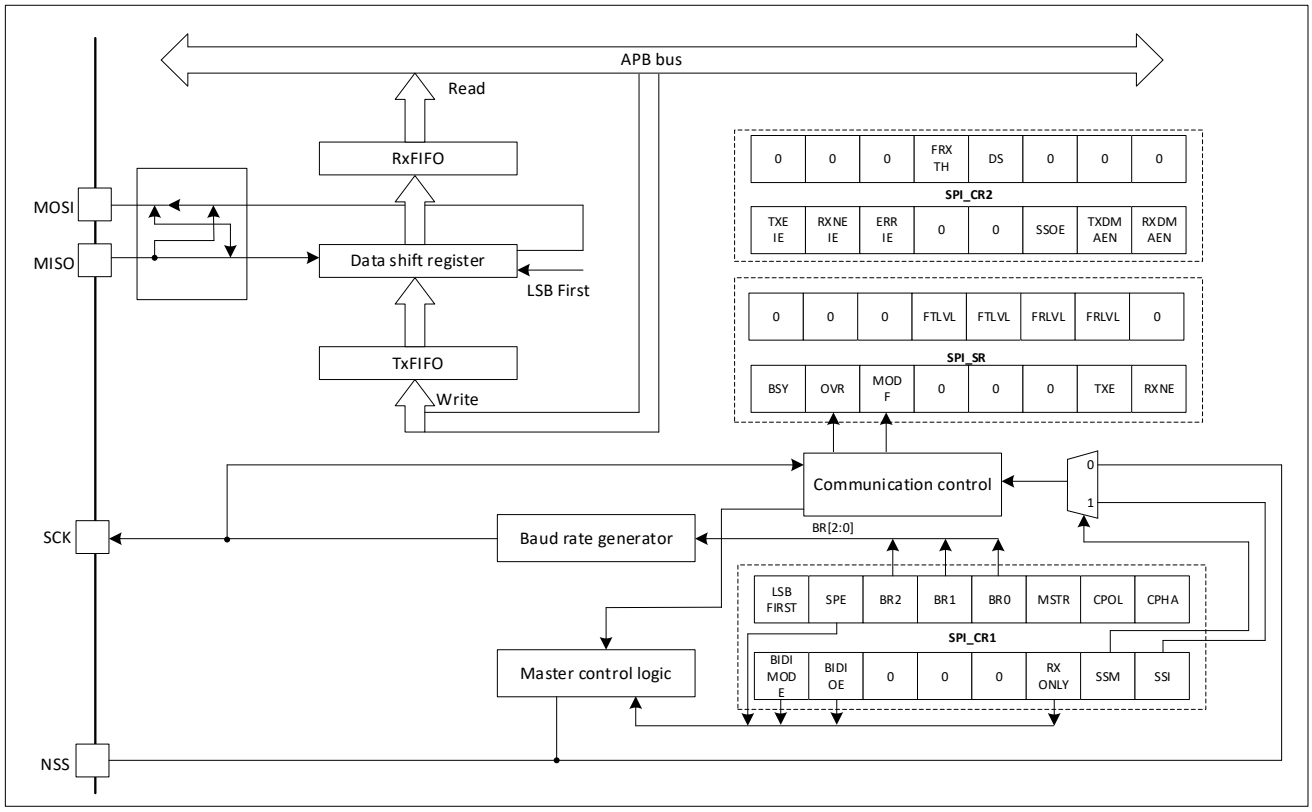

Figure 28-1 SPI block diagram

Four I/O pins are dedicated to SPI communication with external devices:

MISO: Master In / Slave Out data. In the general case, this pin is used to transmit data in slave mode and receive data in master mode.

MOSI: Master Out / Slave In data. In the general case, this pin is used to transmit data in master mode and receive data in slave mode.

SCK: Serial Clock output pin for SPI masters and input pin for SPI slaves.

NSS: Slave select pin. Depending on the SPI and NSS settings, this pin can be used to either:

– Select an individual slave device for communication

– Synchronize the data frame or

– Detect a conflict between multiple masters

The SPI bus allows the communication between one master device and one or more slave devices. The bus consists of at least two wires - one for the clock signal and the other for synchronous data transfer. Other signals can be added depending on the data exchange between SPI nodes and their slave select signal management.

28.3.2. Communications between one master and one slave

The SPI allows the MCU to communicate using different configurations, depending on the device targeted and the application requirements. These configurations use 2 or 3 wires (with software NSS management) or 3 or 4 wires (with hardware NSS management). Communication is always initiated by the master.

28.3.2.1. Full-duplex communication

By default, the SPI is configured for full-duplex communication. In this configuration, the shift registers of the master and slave are linked using two unidirectional lines between the MOSI and the MISO pins. During SPI communication, data is shifted synchronously on the SCK clock edges provided by the master. The master transmits the data to be sent to the slave via the MOSI line and receives data from the slave via the MISO line.

When the data frame transfer is complete (all the bits are shifted) the information between the master and slave is exchanged.

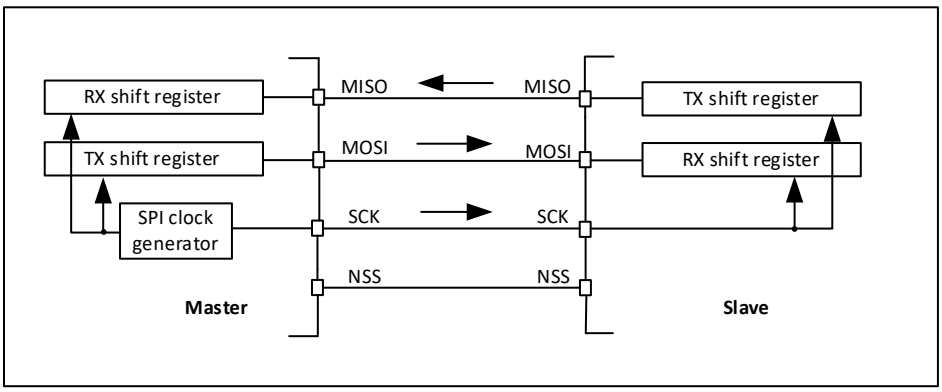

Figure 28-2 Full-duplex single master/slave application

28.3.2.2. Half-duplex communication

The SPI can communicate in half-duplex mode by setting the BIDIMODE bit in the SPIx_CR1 register. In this configuration, one single cross connection line is used to link the shift registers of the master and slave together. During this communication, the data is synchronously shifted between the shift registers on the SCK clock edge in the transfer direction selected reciprocally by both master and slave with the BDIOE bit in their SPIx_CR1 registers. In this configuration, the master's MISO pin and the slave's MOSI pin are free for other application uses and act as GPIOs.

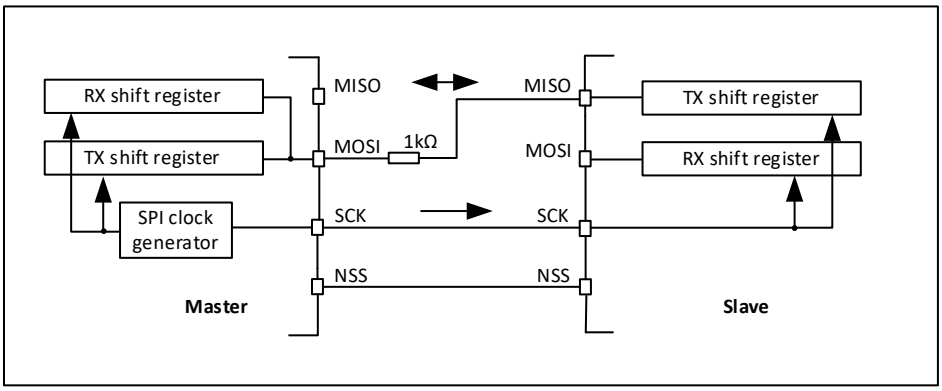

Figure 28-3 Half-duplex single master/slave application

The NSS pins can be used to provide a hardware control flow between master and slave. Optionally, the pins can be left unused by the peripheral. Then the flow has to be handled internally for both master and slave.

28.3.2.3. Simplex communications

The SPI can communicate in simplex mode by setting the SPI in transmit-only or in receiveonly using the RXONLY bit in the SPIx_CR2 register. In this configuration, only one line is used for the transfer between the shift registers of the master and slave. The remaining MISO and MOSI pins pair is not used for communication and can be used as standard GPIOs.

- **Transmit-only mode (RXONLY = 0)**: The configuration settings are the same as for fullduplex. The application has to ignore the information captured on the unused input pin. This pin can be used as a

standard GPIO.

- **Receive-only mode (RXONLY = 1)**: The application can disable the SPI output function by setting the RXONLY bit. In slave configuration, the MISO output is disabled and the pin can be used as a GPIO. The slave continues to receive data from the MOSI pin while its slave select signal is active. Received data events appear depending on the data buffer configuration. In the master configuration, the MOSI output is disabled and the pin can be used as a GPIO. The clock signal is generated continuously as long as the SPI is enabled. The only way to stop the clock is to clear the RXONLY bit or the SPE bit and wait until the incoming pattern from the MISO pin is finished

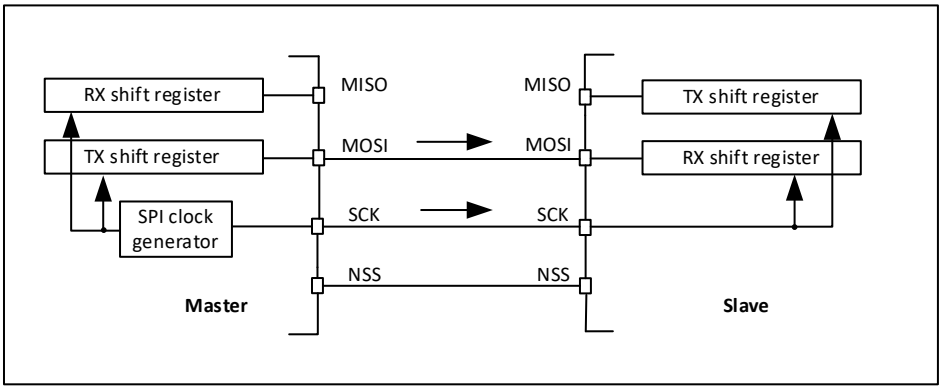

Figure 28-4 simplex single master/single slave application(master in transmit-only/slave in receive-only mode)

(1) The NSS pins can be used to provide a hardware control flow between master and slave. Optionally, the pins can be left unused by the peripheral. Then the flow has to be handled internally for both master and slave. For more details see Section 28.5.5: Slave select (NSS) pin management.

(2) An accidental input information is captured at the input of transmitter Rx shift register. All the events associated with the transmitter receive flow must be ignored in standard transmit only mode.

(3) In this configuration, both the MISO pins can be used as GPIOs.

simplex communication can be replaced by half -duplex communication by setting the transfer direction (the bidirectional mode is enabled when the BI DIO E bit is not changed).

28.3.3. Multi-slave communication

In a configuration with two or more independent slaves, the master uses GPIO to manage NSS for each slave. The master must select a slave by pulling the connected slave NSS low. When this is done, standard master and dedicated slave communication is established.

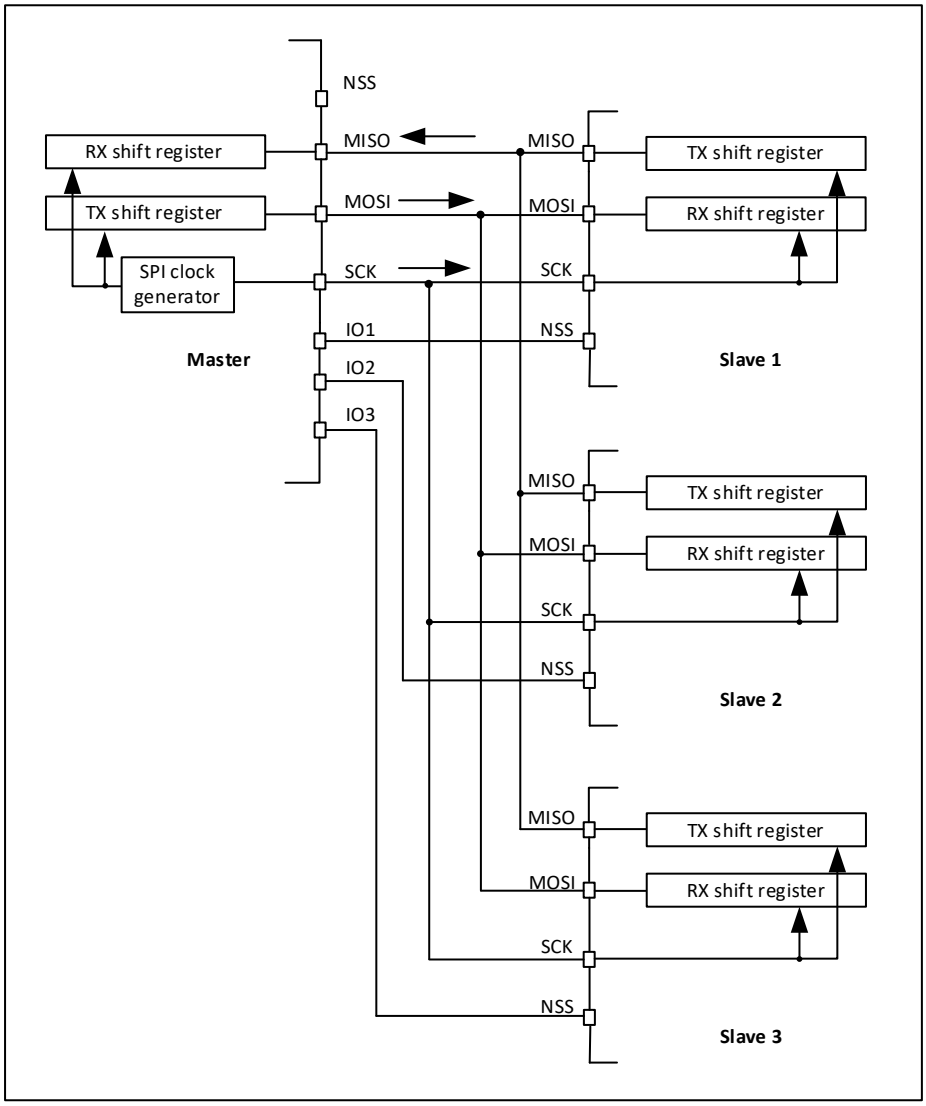

Figure 28-5 Master communicates with three independent slaves

NSS is not used on the host side in this configuration. Any MODF errors must be prevented by SSM = 1, SSI = 1.

Since the MISOs of the slaves are connected together, all slaves must configure their MISO's GPIO as AF open-drain.

28.3.4. Multi-master communication

Unless SPI bus is not designed for a multi-master capability primarily, the user can use build in feature which detects a potential conflict between two nodes trying to master the bus at the same time. For this detection, NSS pin is used configured at hardware input mode. The connection of more than two SPI nodes working at this mode is impossible as only one node can apply its output on a common data line at time.

When nodes are non active, both stay at slave mode by default. Once one node wants to overtake control on the bus, it switches itself into master mode and applies active level on the slave select input of the other node via dedicated GPIO pin. After the session is completed, the active slave select signal is released and the node mastering the bus temporary returns back to passive slave mode waiting for next session start. If potentially both nodes raised their mastering request at the same time a bus conflict event appears (see mode fault MODF

event). Then the user can apply some simple arbitration process (e.g. to postpone next attempt by predefined different time-outs applied at both nodes).

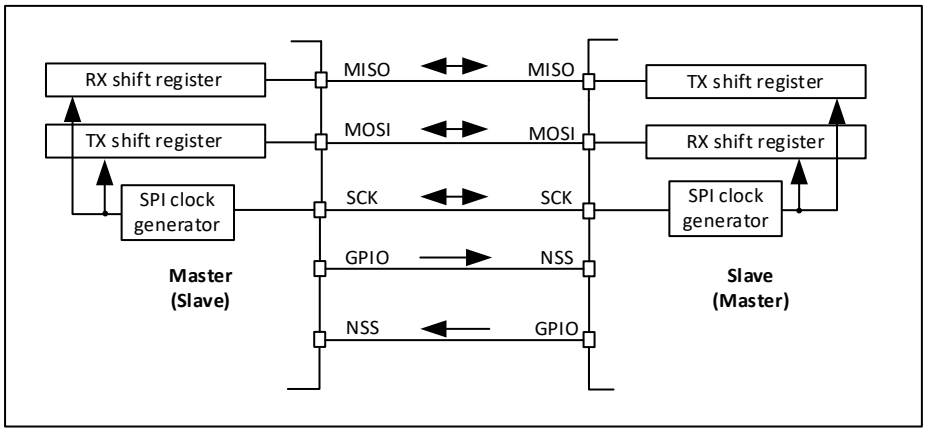

Figure 28-6 Multi-master application

The NSS pin is configured at hardware input mode at both nodes. Its active level enables the MISO line output control as the passive node is configured as a slave.

28.3.5. Slave select (NSS) pin management

In slave mode, the NSS works as a standard “chip select” input and lets the slave communicate with the master. In master mode, NSS can be used either as output or input. s an input it can prevent multimaster bus collision, and as an output it can drive a slave elect signal of a single slave.

Hardware or software slave select management can be set using the SSM bit in the SPIx_CR1 register:

- Software NSS management (SSM = 1): in this configuration, slave select information is driven internally by the SSI bit value in register SPIx_CR1. The external NSS pin is free for other application uses.
- Hardware NSS management (SSM = 0): in this case, there are two possible configurations. The configuration used depends on the NSS output configuration
  1） – NSS output enable (SSM = 0,SSOE = 1): this configuration is only used when the MCU is set as master. The NSS pin is managed by the hardware. The NSS signal is driven low as soon as the SPI is enabled in master mode (SPE = 1), and is kept low until the SPI is disabled (SPE = 0). A pulse can be generated between continuous communications if NSS pulse mode is activated (NSSP = 1). The SPI cannot work in multimaster configuration with this NSS setting.
  2） – NSS output disable (SSM = 0, SSOE = 0): if the microcontroller is acting as the master on the bus, this configuration allows multimaster capability. If the NSS pin is pulled low in this mode, the SPI enters master mode fault state and the device is automatically reconfigured in slave mode. In slave mode, the NSS pin works as a standard “chip select” input and the slave is selected while NSS line is at low level.

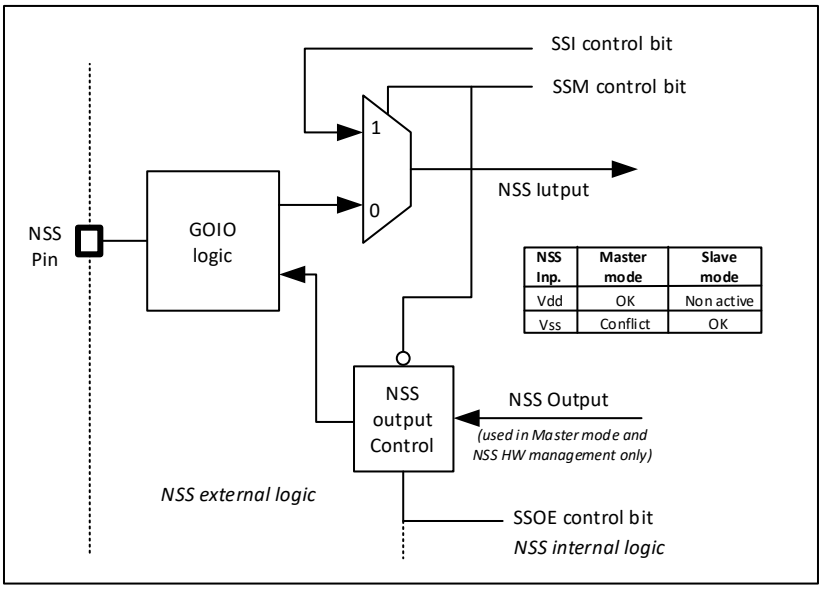

Figure 28-7 Hardware/software slave select management

28.3.6. Communication formats

During SPI communication, receive and transmit operations are performed simultaneously. The serial clock (SCK) synchronizes the shifting and sampling of the information on the data lines. The communication format depends on the clock phase, the clock polarity and the data frame format. To be able to communicate together, the master and slaves devices must follow the same communication format.

28.3.6.1. Clock phase and polarity controls

There are 4 possible timings that can be configured by software through the CPOL and CPHA bits (SPI_CR1 register). CPOL (clock polarity) controls the IDLE state of the clock when no data is being transmitted. This bit affects both master and slave. If CPOL is reset, the SCK pin has a low state. If CPOL is set, the SCK pin has a high IDLE state.

If CPHA is set, the second edge of SCK captures the first data bit transmitted (falling edge if CPOL is reset, rising edge otheRWise). On the occurrence of clock change type, the data is latched. If CPHA is reset, the first edge of SCK captures the first transmitted data bit (falling edge if CPOL is set, rising edge otheRWise). Data is latched when this type of clock change occurs.

The combination of CPOL and CPHA selects the data capture clock edge.

SPI must be disabled (SPE = 0) before CPOL/CPHA is changed.

The IDLE state of SCK must correspond to the polarity selected by the SPI_CR1 register.

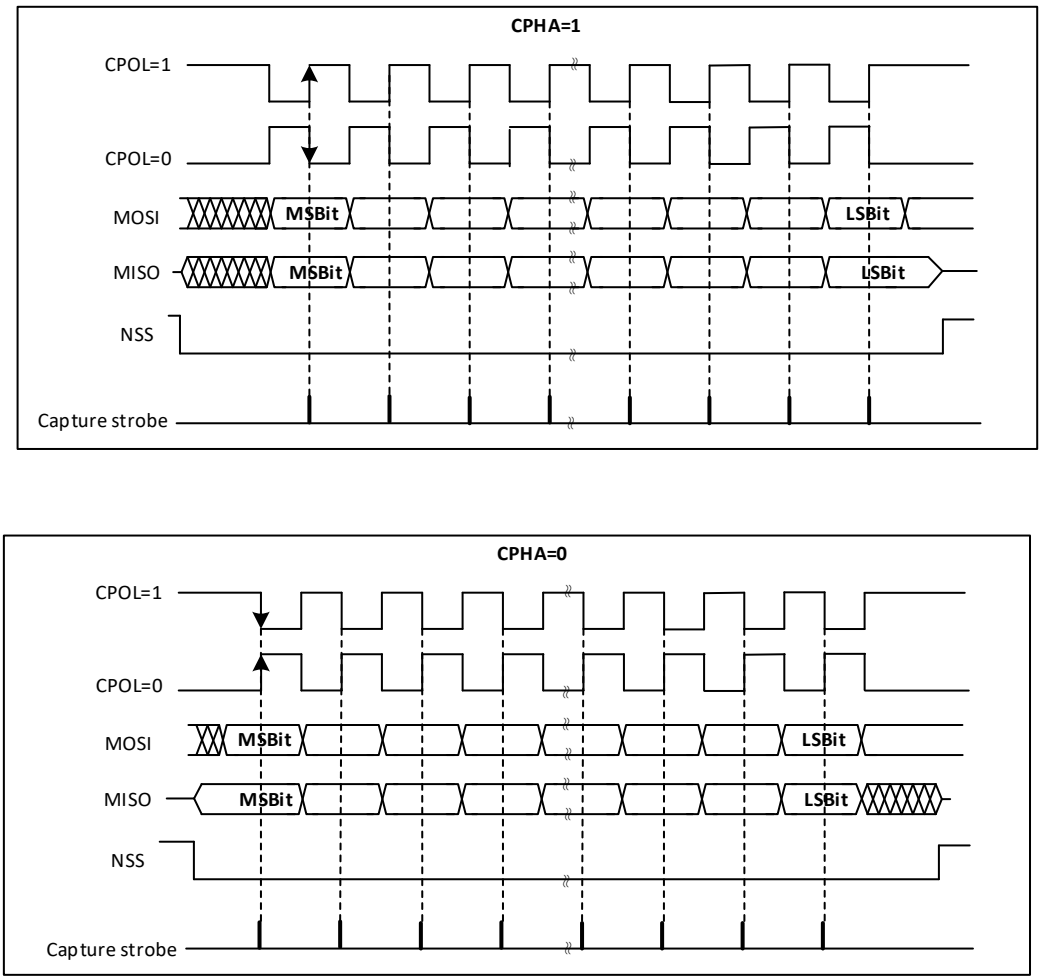

Figure 28-8 Data clock timing diagram

The order of data bits depends on LSBFIRST bit setting.

28.3.6.2. Data frame format

Through the LSBFIRST bit (SPI_CR 1 register), the SPI sh ift register can be set to MSB-FIRST or LSB-FIRST. Select the number of bits in the data frame by using the DS bit (SPI_CR2 register). It can be selected as 8-bit or 16-bit length, and this setting applies to both sending and receiving.

28.3.7. SPI configuration

The configuration procedure is almost the same for master and slave. For specific mode setups, follow the dedicated sections. When a standard communication is to be initialized, perform these steps:

1. Write related GPIO registers: configure MOSI, MISO and SCK pins
2. Write SPI_CR1 register
   1） Configure the clock baud rate via BR[2:0] (not required for slave mode)
   2） Configure CPOL and CPHA
   3） simplex or half -duplex mode by RXONLY or B IDIMODE and BIDIOE (RXONLY and BIDIMODE cannot be active at the same time)
   4） Configure LSBFIRST
   5） Configure SSM and SSI
   6） Configure the MSTR bit (in multi-master In NSS configuration, if the host is configured to prevent MODF

errors, avoid NSS conflict state)

3. Write SPI_CR2 register
   1） Configure DS bit, select the number of data frame bits
   2） Configure SSOE (not required for slave mode)
   3） Configure the FRXTH bit. RXFIFO thresholds must be aligned with the number of bits accessed to the SPI_DR register
4. Write the corresponding DMA register: configure the SPI Tx and Rx channels of the DMA

28.3.8. Procedure for enabling SPI

It is recommended to enable the SPI slave before the master sends the clock. If not, undesired data transmission might occur. The data register of the slave must already contain data to be sent before starting communication with the master (either on the first edge of the communication clock, or before the end of the ongoing communication if the clock signal is continuous). The SCK signal must be settled at an idle state level corresponding to the selected polarity before the SPI slave is enabled.

The master at full-duplex (or in any transmit-only mode) starts to communicate when the SPI is enabled and TXFIFO is not empty, or with the next write to TXFIFO.

In any master receive only mode (RXONLY = 1 or BIDIMODE = 1 & BIDIOE = 0), master starts to communicate and the clock starts running immediately after SPI is enabled.

For handling DMA, follow the dedicated section.

28.3.9. Data transmission and reception procedures

28.3.9.1. RXFIFO and TXFIFO

All SPI data transactions pass through the 32-bit embedded FIFOs. This enables the SPI to work in a continuous flow, and prevents overruns when the data frame size is short. Each direction has its own FIFO called TXFIFO and RXFIFO. These FIFOs are used in all SPI modes except for receiver-only mode (slave or master).

The handling of FIFOs depends on the data exchange mode (duplex, simplex), data frame format (number of bits in the frame), access size performed on the FIFO data registers (8-bit or 16-bit).

A read access to the SPIx_DR register returns the oldest value stored in RXFIFO that has not been read yet. A write access to the SPIx_DR stores the written data in the TXFIFO at the end of a send queue. The read access must be always aligned with the RXFIFO threshold configured by the FRXTH bit in SPIx_CR2 register. FTLVL[1:0] and FRLVL[1:0] bits indicate the current occupancy level of both FIFOs.

A read access to the SPIx_DR register must be managed by the RXNE event. This event is triggered when data is stored in RXFIFO and the threshold (defined by FRXTH bit) is reached. When RXNE is cleared, RXFIFO is considered to be empty.

In a similar way, write access of a data frame to be transmitted is managed by the TXE event. This event is triggered when the TXFIFO level is less than or equal to half of its capacity. OtheRWise TXE is cleared and the TXFIFO is considered as full.

In this way, RXFIFO can store up to four data frames, whereas TXFIFO can only store up to three when the data frame format is not greater than 8 bits. This difference prevents possible corruption of 3x 8-bit data frames already stored in the TXFIFO when software tries to write more data in 16-bit mode into TXFIFO.

Both TXE and RXNE events can be polled or handled by interrupts.

Another way to manage the data exchange is to use DMA.

If the next data is received when the RXFIFO is full, an overrun event occurs. An overrun event can be polled or handled by an interrupt.

The BSY bit being set indicates ongoing transaction of a current data frame. When the clock signal runs continuously, the BSY flag stays set between data frames at master but becomes low for a minimum duration of one SPI clock at slave between each data frame transfer.

28.3.9.2. Sequence handling

A few data frames can be passed at single sequence to complete a message. When transmission is enabled, a sequence begins and continues while any data is present in the TXFIFO of the master. The clock signal is provided continuously by the master until TXFIFO becomes empty, then it stops waiting for additional data.

In receive-only modes, half-duplex (BIDIMODE = 1, BIDIOE = 0) or simplex (BIDIMODE = 0, RXONLY = 1) the master starts the sequence immediately when both SPI is enabled and receive-only mode is activated. The clock signal is provided by the master and it does not stop until either SPI or receive-only mode is disabled by the master. The master receives data frames continuously up to this moment.

While the master can provide all the transactions in continuous mode (SCK signal is continuous) it has to respect slave capability to handle data flow and its content at anytime. When necessary, the master must slow down the communication and provide either a slower clock or separate frames or data sessions with sufficient delays. Be aware there is no underflow error signal for master or slave in SPI mode, and data from the slave is always transacted and processed by the master even if the slave could not prepare it correctly in time. It is preferable for the slave to use DMA, especially when data frames are shorter and bus rate is high.

Each sequence must be encased by the NSS pulse in parallel with the multislave system to select just one of the slaves for communication. In a single slave system it is not necessary to control the slave with NSS, but it is often better to provide the pulse here too, to synchronize the slave with the beginning of each data sequence. NSS can be managed by both software and hardware (see Section 28.5.5: Slave select (NSS) pin management).

When the BSY bit is set it signifies an ongoing data frame transaction. When the dedicated frame transaction is finished, the RXNE flag is raised. The last bit is just sampled and the complete data frame is stored in the RXFIFO.

28.3.9.3. Procedure for disabling the SPI

When SPI is disabled, it is mandatory to follow the disable procedures described in this paragraph. It is important to do this before the system enters a low-power mode when the peripheral clock is stopped. Ongoing transactions can be corrupted in this case. In some modes the disable procedure is the only way to stop continuous communication running.

Master in full-duplex or transmit only mode can finish any transaction when it stops providing data for transmission. In this case, the clock stops after the last data transaction. Special care must be taken in packing mode when an odd number of data frames are transacted to prevent some dummy byte exchange (refer to Data packing section). Before the SPI is disabled in these modes, the user must follow standard disable procedure.

When the SPI is disabled at the master transmitter while a frame transaction is ongoing or next data frame is stored in TXFIFO, the SPI behavior is not guaranteed.

When the master is in any receive only mode, the only way to stop the continuous clock is to disable the peripheral by SPE = 0. Specific procedure must be followed when disabling SPI in this mode.

Data received but not read remains stored in RXFIFO when the SPI is disabled, and must be processed the next time the SPI is enabled, before starting a new sequence. To prevent having unread data, ensure that RXFIFO is empty when disabling the SPI, by using the correct disabling procedure, or by initializing all the SPI registers with a software reset via the control of a specific register dedicated to peripheral reset.

Standard disable procedure is based on pulling BSY status together with FTLVL[1:0] to check if a transmission session is fully completed. This check can be done in specific cases, too, when it is necessary to identify the end of ongoing transactions, for example:

- When NSS signal is managed by software and master has to provide proper end of NSS pulse for slave.
- When transactions' streams from DMA or FIFO are completed while the last data frame or CRC frame transaction is still ongoing in the peripheral bus.

The correct disable procedure is (except when receive only mode is used):

1. Wait until FTLVL[1:0] = 00 (no more data to transmit).
2. Wait until BSY = 0 (the last data frame is processed).
3. Disable the SPI (SPE = 0).
4. Read data until FRLVL[1:0] = 00 (read all the received data).

The correct disable procedure for certain receive only modes is:

1. Interrupt the receive flow by disabling SPI (SPE = 0) in the specific time window while the last data frame is ongoing.
2. Wait until BSY = 0 (the last data frame is processed).
3. Read data until FRLVL[1:0] = 00 (read all the received data).

28.3.9.4. Data packing

When the data frame size fits into one byte (equal to 8 bits), data packing is used automatically when any read or write 16-bit access is performed on the SPIx_DR register. The double data frame pattern is handled in parallel in this case. First, SPI uses the pattern stored in the low-order bits of the word being accessed, followed by the high-order bits.

The figure below provides the data packing process. After a single 16- bit access by the sender, two data frames are sent. On the receiver side, if the RXFIFO threshold is set to 16 bits (FRXTH = 0), the sequence will be generated immediately on the RXNE event. In response to the RXNE event, the receiver accesses 2 data frames through a 16-bit read of the SPI_DR register. On the receiver side, the setting of the RxFIFO threshold and subsequent read accesses must remain aligned, otheRWise data will be lost.

On the sender side, it is sufficient to write the last data frame of the odd sequence with 8- bit access. In order to generate an RxNE event, for an odd number of data frames, the receiver must change the Rx_FIFO threshold for the last data frame received.

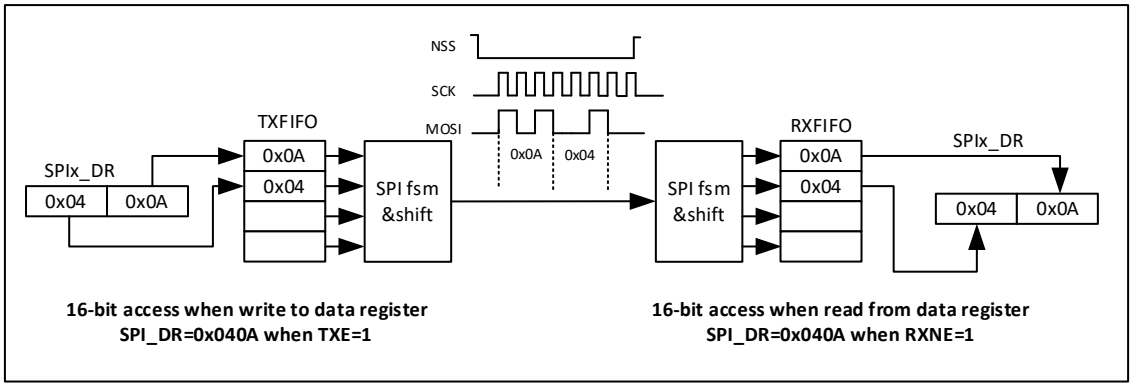

Figure 28-9 Packing data in FIFO for transmission and reception

28.3.9.5. Communication using DMA

To operate at its maximum speed and to facilitate the data register read/write process required to avoid overrun, the SPI features a DMA capability, which implements a simple request/acknowledge protocol.

A DMA access is requested when the TXE or RXNE enable bit in the SPIx_CR2 register is set. Separate requests must be issued to the Tx and Rx buffers.

- In transmission, a DMA request is issued each time TXE is set to 1. The DMA then writes to the SPIx_DR register.
- In reception, a DMA request is issued each time RXNE is set to 1. The DMA then reads the SPIx_DR register.

When the SPI is used only to transmit data, it is possible to enable only the SPI Tx DMA channel. In this case, the OVR flag is set because the data received is not read. When the SPI is used only to receive data, it is possible to enable only the SPI Rx DMA channel.

In transmission mode, when the DMA has written all the data to be transmitted (the TCIF flag is set in the DMA_ISR register), the BSY flag can be monitored to ensure that the SPI communication is complete. This is required to avoid corrupting the last transmission before disabling the SPI or entering the Stop mode. The software must first wait until FTLVL[1:0] = 00 and then until BSY = 0.

When starting communication using DMA, to prevent DMA channel management raising error events, these steps must be followed in order:

1. Enable DMA Rx buffer in the RXDMAEN bit in the SPI_CR2 register, if DMA Rx is used.
2. Enable DMA streams for Tx and Rx in DMA registers, if the streams are used.
3. Enable DMA Tx buffer in the TXDMAEN bit in the SPI_CR2 register, if DMA Tx is used.
4. Enable the SPI by setting the SPE bit.

To close communication it is mandatory to follow these steps in order:

1. Disable DMA streams for Tx and Rx in the DMA registers, if the streams are used.
2. Disable the SPI by following the SPI disable procedure.
3. Disable DMA Tx and Rx buffers by clearing the TXDMAEN and RXDMAEN bits in the SPI_CR2 register, if DMA Tx and/or DMA Rx are used.

28.3.9.6. Packing with DMA

If the transfers are managed by DMA (TXDMAEN and RXDMAEN set) packing mode is enabled/disabled automatically depending on the PSIZE value configured for SPI TX and the SPI RX DMA channel. If the DMA channel PSIZE value is equal to 16-bit and SPI data size is equal to 8-bit, then packing mode is enabled. The DMA then automatically manages the write operations to the SPIx_DR register.

If data packing mode is used and the number of data to transfer is not a multiple of two, the LDMA_TX/LDMA_RX bits must be set. The SPI then considers only one data for the transmission or reception to serve the last DMA transfer.

28.3.9.7. Communication diagrams

Some typical timing schemes are explained in this section. These schemes are valid no matter if the SPI events are handled by polling, interrupts or DMA. For simplicity, the LSBFIRST = 0, CPOL = 0 and CPHA = 1 setting is used as a common assumption here. No complete configuration of DMA streams is provided.

1. The slave starts to control MISO line as NSS is active and SPI is enabled, and is disconnected from the line when one of them is released. Sufficient time must be provided for the slave to prepare data dedicated to the master in advance before its transaction starts.At the master, the SPI peripheral takes control at MOSI and SCK signals (occasionally at NSS signal as well) only if SPI is enabled. If SPI is disabled the SPI peripheral is disconnected from GPIO logic, so the levels at these lines depends on GPIO setting exclusively.
2. At the master, BSY stays active between frames if the communication (clock signal) is continuous. At the slave, BSY signal always goes down for at least one clock cycle between data frames.
3. The TXE signal is cleared only if TXFIFO is full.
4. The DMA arbitration process starts just after the TXDMAEN bit is set. The TXE interrupt is generated just after the TXEIE is set. As the TXE signal is at an active level, data transfers to TxFIFO start, until TxFIFO becomes full or the DMA transfer completes.
5. If all the data to be sent can fit into TxFIFO, the DMA Tx TCIF flag can be raised even before communication on the SPI bus starts. This flag always rises before the SPI transaction is completed.
6. In Data packed mode, Tx E and R xNE events are paired, and each read/write FIFO access is 16 bits wide (until the number of data frames are even). If TxFIFO is 3/4 full, the F TLVL state stops at F IFO full level. That's why the last odd frame cannot be stored before the TxFIFO becomes 1/2 full. The data frame is stored in TxFIFO in 8 -bit access mode (software or DMA automatic access when L DMA_TX control is set).
7. To receive the last odd data frame in packed mode, the Rx threshold must be changed to 8 -bit when the last data frame is processed (either by software setting FRXTH = 1, or when LDMA_RX is Automatic processing of DMA internal signals when set).

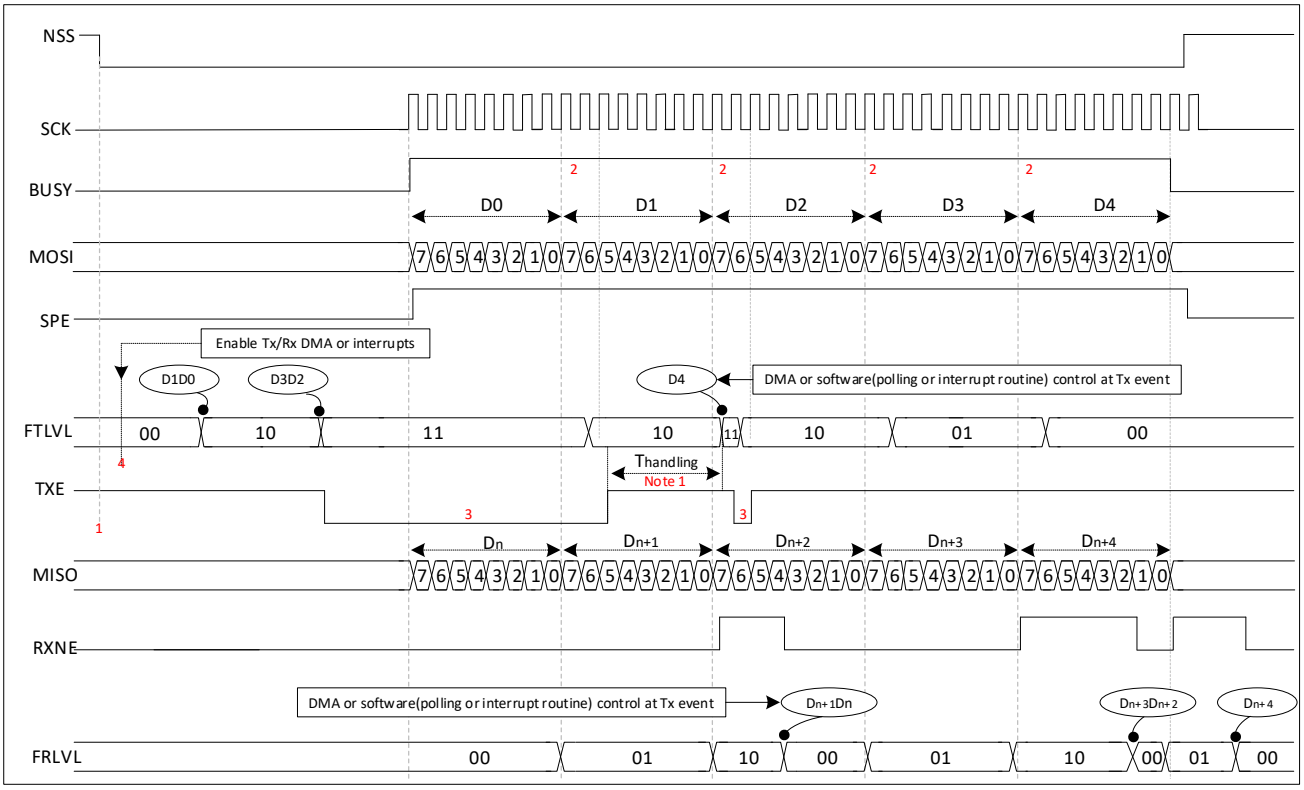

Figure 28-10 Master full-duplex communication diagram(bit frame = 8, FRXTH = 0)

Note1: $T_{handling}$ means cpu writes data to Tx fifo time spent.

28.3.10. Status flags

Three status flags are provided for the application to completely monitor the state of the SPI bus.

28.3.10.1. Tx buffer empty flag (TXE)

The TXE flag is set when transmission TXFIFO has enough space to store data to send. TXE flag is linked to the TXFIFO level. The flag goes high and stays high until the TXFIFO level is lower or equal to 1/2 of the FIFO depth. An interrupt can be generated if the TXEIE bit in the SPIx_CR2 register is set. The bit is cleared automatically when the TXFIFO level becomes greater than 1/2.

28.3.10.2. Rx buffer not empty (RXNE)

The RXNE flag is set depending on the FRXTH bit value in the SPIx_CR2 register:

- If FRXTH is set, RXNE goes high and stays high until the RXFIFO level is greater or equal to 1/4 (8-bit).
- If FRXTH is cleared, RXNE goes high and stays high until the RXFIFO level is greater than or equal to 1/2 (16-bit).

An interrupt can be generated if the RXNEIE bit in the SPIx_CR2 register is set.

The RXNE is cleared by hardware automatically when the above conditions are no longer true.

28.3.10.3. Busy flag (BSY)

The BSY flag is set and cleared by hardware (writing to this flag has no effect). This flag indicates the state of the SPI communication layer.

When it is set to '1', it indicates that the SPI is busy communicating, with one exception: in the bidirectional receive mode of master mode (MSTR = 1, BDM = 1 and BDOE = 0), the BSY flag is held during reception to low.

The BSY flag can be used in certain modes to detect the end of a transfer so that the software can disable the SPI or its peripheral clock before entering a low-power mode which does not provide a clock for the peripheral. This avoids corrupting the last transfer, so it needs to strictly follow the procedure below.

The BSY flag is also useful for preventing write collisions in a multimaster system.

Except for the bidirectional receive mode of the master mode (MSTR = 1, BDM = 1 and BDOE = 0), the BSY flag is set to '1' when the transmission starts.

The BSY flag is cleared under any one of the following conditions:

● When the SPI is correctly disabled

● When a fault is detected in Master mode (MODF bit set to 1)

● In Master mode, when it finishes a data transmission and no new data is ready to be sent

● In Slave mode, when the BSY flag is set to '0' for at least one SPI clock cycle between each data transfer.

Note: it is recommended to use always the TXE and RXNE flags (instead of the BSY flags) to handle data transmission or reception operations.

28.3.11. Error flags

28.3.11.1. Mode fault (MODF)

Mode fault occurs when the master device has its internal NSS signal (NSS pin in NSS hardware mode, or SSI bit in NSS software mode) pulled low. This automatically sets the MODF bit. Master mode fault affects the SPI interface in the following ways:

● The MODF bit is set and an SPI interrupt is generated if the ERRIE bit is set.

● The SPE bit is cleared. This blocks all output from the device and disables the SPI interface.

● The MSTR bit is cleared, thus forcing the device into slave mode.

Use the following software sequence to clear the MODF bit:

1. Make a read or write access to the SPIx_SR register while the MODF bit is set.

2. Then write to the SPIx_CR1 register.

To avoid any multiple slave conflicts in a system comprising several MCUs, the NSS pin must be pulled high during the MODF bit clearing sequence. The SPE and MSTR bits can be restored to their original state after this clearing sequence. As a security, hardware does not allow the SPE and MSTR bits to be set while the MODF bit is set. In a slave device the MODF bit cannot be set except as the result of a previous multimaster conflict.

28.3.11.2. Overrun flag (OVR)

An overrun condition occurs when data is received by a master or slave and the RXFIFO has not enough space to store this received data. This can happen if the software or the DMA did not have enough time to read the previously received data (stored in the RXFIFO) or when space for data storage is limited.

When an overrun condition occurs, the newly received value does not overwrite the previous one in the RXFIFO. The newly received value is discarded and all data transmitted subsequently is lost.

Clearing the OVR bit is done by a read access to the SPI_DR register followed by a read access to the SPI_SR register.

28.3.12. SPI interrupts

Table 28-1 SPI interrupt requests

Interrupt event	Event flag	Enable Control bit

Transmit TXFIFO ready to be loaded	TXE	TXEIE
Data received in RXFIFO	RXNE	RXNEIE
Master Mode fault event	MODF	ERRIE
Overrun error	OVR	ERRIE

28.4. SPI register

The SPI registers have to be accessed by 16 bits and 32 bits, and the DR registers support 32 bits, 16 bits and 8 bits.

28.4.1. SPI control register 1 (SPI_CR1)

Address offset: 0x00

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BI-DIMODE	BIDIOE	Res	Res	Res	RXONLY	SSM	SSI	LSBFIRST	SPE	BR[2:0]			MSTR	CPOL	CPHA
RW	RW				RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
15	BIDIMODE	RW	0	Bidirectional data mode enable 0: 2-line unidirectional data mode 1: 1-line bidirectional data mode
14	BIDIOE	RW	0	Output enable in bidirectional mode This bit combined with the BIDIMODE bit selects the direction of transfer in bidirectional mode 0: Output disabled (receive-only mode) 1: Output enabled (transmit-only mode) In master mode, the MOSI pin is used and in slave mode, the MISO pin is used.
13	Reserved		0	
12	Reserved		0	
11	Reserved	RES	-	Reserved
10	RXONLY	RW	0	Receive control only. This bit, together with the BIDIMODE bit, determines the transfer direction in "2-wire unidirectional" mode. In the configuration of multiple slave devices, this bit is set to 1 on the slave device that is not accessed, so that only the slave device that is accessed has output, so there will be no data conflict on the data line. 0: Full-duplex (Transmit and receive) 1: Output disabled (Receive-only mode)
9	SSM	RW	0	Software slave management When the SSM bit is set, the NSS pin input is replaced with the value from the SSI bit. 0: Software slave management disabled 1: Software slave management enabled
8	SSI	RW	0	Internal slave select This bit has an effect only when the SSM bit is set. The value of this bit is forced onto the NSS pin and the I/O value of the NSS pin is ignored.
7	LSBFIRST	RW	0	Frame format 0: Data is transmitted with the MSB first. 1: Data is transmitted with the LSB first. This bit should not be changed when communication is ongoing
6	SPE	RW	0	SPI enable 0: SPI disabled 1: SPI enable
5:3	BR[2:0]	RW	0	Baud rate control 000: $f_{PCLK}/2$

				001: f_{PCLK}/4 010: f_{PCLK}/8 011: f_{PCLK}/16 100: f_{PCLK}/32 101: f_{PCLK}/64 110: f_{PCLK}/128 111: f_{PCLK}/256 These bits should not be changed when communication is ongoing. Note: In slave mode, the fastest baud rate only supports f_{PCLK}/4.
2	MSTR	RW	0	Master selection 0: Slave configuration 1: Master configuration Note: This bit should not be changed when communication is ongoing.
1	CPOL	RW	0	Clock polarity 0: CK to 0 when idle 1: CK to 1 when idle This bit should not be changed when communication is ongoing.
0	CPHA	RW	0	Clock phase 0: The first clock transition is the first data capture edge 1: The second clock transition is the first data capture edge This bit should not be changed when communication is ongoing.

28.4.2. SPI control register 2 (SPI_CR2)

Address offset: 0x04

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SLVFM	LDMA_RX	LDMA_TX	FRXTH	DS	Res	Res	Res	TXEIE	RXNEIE	ERRIE	Res	Res	SSOE	TXDMAEN	RXDMAEN
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW			RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved		-	Reserved
15	SLVFM	RW	0	Slave fast mode enable 0: Slave normal mode, the speed of the slave mode supporting the fastest SPI clock is less than pclk/4 1: Slave fast mode, can support SPI clock speed in slave mode up to pclk/4 Note: When the speed of SPI clock is less than pclk/4, this register bit must not be set.
14	LDMA_TX	RW	0	Last DMA transfer for transmission This bit is used to define whether the total number of DMA transmissions is even or odd. This function is only valid when data length = 8, the TXDMAEN bit is set, and a 16-bit write access is available. This bit needs to be written when SPE = 0. 0: Number of data to transfer is even 1: Number of data to transfer is odd
13	LDMA_RX	RW	0	Last DMA transfer for transmission This bit is used to define whether the total number of DMA receptions is even or odd. This function is only valid when data length = 8, the RXDMAEN bit is set, and a 16-bit write access is available. This bit needs to be written when SPE = 0. 0: Number of data to transfer is even 1: Number of data to transfer is odd

12	FRXTH	RW	0	FIFO reception threshold This bit is used to set the threshold of the RXFIFO that triggers an RXNE event 0: RXNE event is generated if the FIFO level is greater than or equal to 1/2 (16-bit) 1: RXNE event is generated if the FIFO level is greater than or equal to 1/4 (8-bit)
11	DS	RW	0	SPI transmission data length 0: 8-bit data frame transmission 1: 16-bit data frame transmission
10:8	Reserved			
7	TXEIE	RW	0	Tx buffer empty interrupt enable 0: TXE interrupt masked 1: TXE interrupt not masked. Used to generate an interrupt request when the TXE flag is set
6	RXNEIE	RW	0	RX buffer not empty interrupt enable 0: RXNE interrupt masked 1: RXNE interrupt not masked. Used to generate an interrupt request when the RXNE flag is set
5	ERRIE	RW	0	Error interrupt enable 0: Error interrupt is masked 1: Error interrupt is enabled This bit controls the generation of an interrupt when an error condition occurs (CRCERR, VR, MODF in SPI mode).
4:3	Reserved	RES	-	Reserved
2	SSOE	RW	0	SS output enable 0: SS output is disabled in master mode and the SPI interface can work in multimaster configuration 1: SS output is enabled in master mode and when the SPI interface is enabled. The SPI interface cannot work in a multimaster environment.
1	TXDMAEN	RW	0	Tx buffer DMA enable 0: Tx buffer DMA disabled 1: Tx buffer DMA enabled. When this bit is set, a DMA request is generated whenever the TXE flag is set.
0	RXDMAEN	RW	0	Rx buffer DMA enable 0: Rx buffer DMA disabled 1: Rx buffer DMA enabled. When this bit is set, a DMA request is generated whenever the RXNE flag is set.

Note :

There are a total of 4 combinations of FRXTH and DS, but the process of restricting the use of the software is as follows:

1. This bit should be 0 if DS = F is configured (ie the transmission data length is 16)
2. If DS = 7 is configured (that is, the transmission data length is 8), the following two situations need to be distinguished:
   1） If the number of data frames of communication is 1 frame of data, you need to set this bit to 1
   2） If the number of communication data frames is greater than 1 frame data, clear this bit to 0

28.4.3. SPI status register (SPI_SR)

Address offset: 0x08

Reset value: 0x0002

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	FTLVL [1:0]		FRLVL [1:0]		Res	BSY	OVR	MODF	Res	Res	Res	TXE	RXNE
			R	R	R	R		R	R	R				R	R

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	RES	-	Reserved
12:11	FTLVL	R	0	FIFO Transmission Level These bits are set and cleared by hardware. 00: FIFO empty 01: 1/4 FIFO 10: 1/2 FIFO 11: FIFO full (considered as FULL when the FIFO threshold is greater than 1/2)
10:9	FRLVL	R	0	FIFO reception level These bits are set and cleared by hardware. 00: FIFO empty 01: 1/4 FIFO 10: 1/2 FIFO 11: FIFO full
7	BSY	R	0	Busy flag *0:* SPI (or I2S) not busy 1: SPI (or I2S) is busy in communication or Tx buffer is not empty
6	OVR	R	0	Overrun flag 0: No overrun occurred 1: Overrun occurred This flag is set by hardware and reset by a software sequence.
5	MODF	R	0	Mode fault 0: No mode fault occurred 1: Mode fault occurred This flag is set by hardware and reset by a software sequence.
4:2	Reserved		0	
1	TXE	R	1	Transmit buffer empty 0: Tx buffer not empty 1: Tx buffer empty
0	RXNE	R	0	Receive buffer not empty 0: Rx buffer empty 1: Rx buffer not empty

28.4.4. SPI data register (SPI_DR)

Address offset: 0x0C

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DR[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
15:0	DR[15:0]	RW	0	Data register Data received or to be transmitted The data register serves as an interface between the Rx and Tx FIFOs. When the data register is read, RxFIFO is accessed while the write to data register accesses TxFIFO. Note:Depending on the DS bit (data frame width selection), data transmission or reception is 8-bit or 16-bit. For 8-bit data frames, the data registers are sent and received based on right -aligned 8-bit data. When in receive mode, DR [15:8] is set to 0 by hardware. For 16-bit data frame, the data register is 16-bit, and the entire DR [15:0] is used for transmit and receive.

28.4.5. SPI register map

Offset	Register	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

0x00	SPI_CR1	BI-DIMODE	BIDIOE	Res.	Res.	Res.	RXONLY	SSM	SSI	LSBFIRST	SPE	BR[2:0]			MSTR	CPOL	CPHA
	Reset value	0	0				0	0	0	0	0	0	0	0	0	0	0
0x04	SPI_CR2	SLVFM	LDMA_RX	LDMA_TX	FRXTH	DS	Res.	Res.	Res.	TXEIE	RXNEIE	ERRIE	Res.	Res.	SSOE	TXDMAEN	RXDMAEN
	Reset value	0	0	0	0	0				0	0	0			0	0	0
0x08	SPI_SR	Res.	Res.	Res.	FTLVL[1:0]		FRLVL[1:0]		Res.	BSY	OVR	MODEF	Res.	Res.	Res.	TXE	RXNE
	Reset value				0	0	0	0		0	0	0				1	0
0x0C	SPI_DR	DR[15:0]															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

29. Debug support

29.1. Overview

This devices are built around a Cortex-M0+ core which contains hardware extensions for advanced debugging features. The debug extensions allow the core to be stopped either on a given instruction fetch (breakpoint) or data access (watchpoint). When stopped, the core's internal state and the system's external state may be examined. Once examination is complete, the core and the system may be restored and program execution resumed.

The debug features are used by the debugger host when connecting to and debugging the MCUs.One interface for debug is available: serial wire. The debugging function in M0+ CPU Core is a set of ARM CoreSight Design kit.

The ARM Cortex®-M0 core provides integrated on-chip debug support. It is comprised of:

- SW-DP: serial wire
- BPU: Break point unit
- DWT: Data watchpoint trigger

It also includes debug features dedicated to this chip:

- Flexible debug pinout assignment
- MCU debug box (support for low-power modes, control over peripheral clocks, etc.)

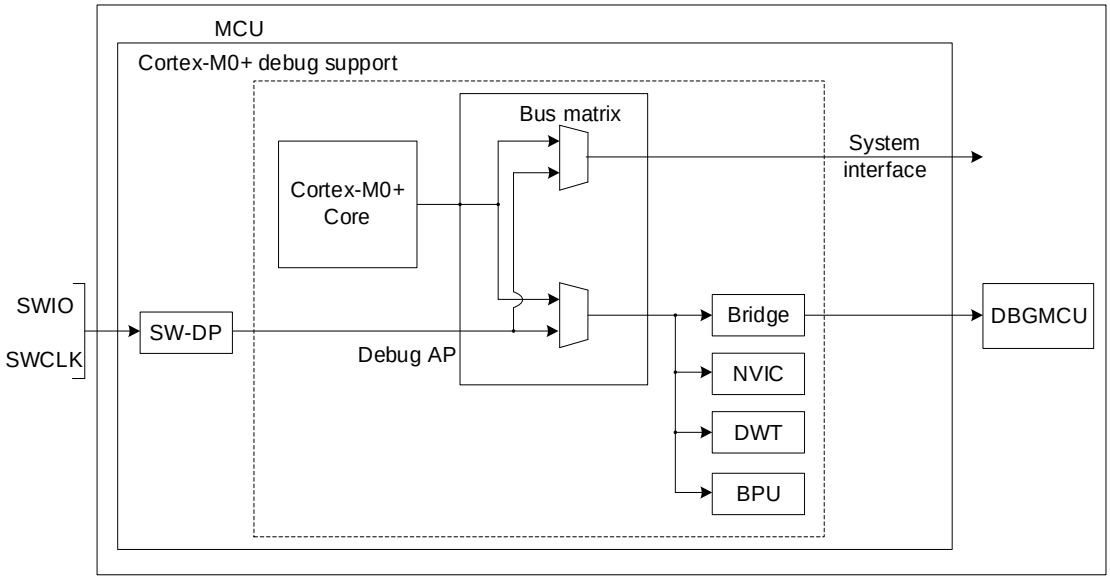

Figure 29-1 Block diagram of MCU debug support

29.2. Pinout and debug port pins

29.2.1. SWD port pins

Two pins are used to debug, these pins are available on all packages.

Table 29-1 SWD port pins

SW-DP Pin name	SW debug port		Pin assignment
	Type	Debug assignment	
SWDIO	I/O	Serial Wire Data Input/Output	PA13

SWDCLK	I	Serial Wire Clock	PA14

29.2.2. SW-DP pin assignment

After reset (SYSRESETn or PORESETn), the pins used for the SW-DP are assigned as dedicated pins which are immediately usable by the debugger host.

However, the MCU offers the possibility to disable the SWD port and can then release the associated pins for general-purpose I/O (GPIO) usage.

29.2.3. Internal pull-up & pull-down on SWD pins

Once the SW I/O is released by the user software, the GPIO controller takes control of these pins. The reset states of the GPIO control registers put the I/Os in the equivalent states:

- SWDIO: input pull-up
- SWCLK: input pull-down

Having embedded pull-up and pull-down resistors removes the need to add external resistors.

29.3. ID codes and locking mechanism

here are several ID codes inside the MCU. It is recommended that Keil, IAR and other tools use this ID Code (located at 0x4001 5800) locks debugging.

After the chip is powered on, the hardware reads the 0x1FFF 0FF8 address of the flash 's factory config.byte and loads it into the DBG _IDCODE register.

29.4. SWD debug port

29.4.1. SWD protocol introduction

This synchronous serial protocol uses two pins:

- SWCLK: clock from host to target
- SWDIO: bidirectional

The protocol allows two banks of registers (DPACC registers and APACC registers) to be read and written to. Bits are transferred LSB-first on the wire. For SWDIO bidirectional management, the line must be pulled-up on the board (100 kΩ recommended by ARM).

Each time the direction of SWDIO changes in the protocol, a turnaround time is inserted where the line is not driven by the host nor the target. By default, this turnaround time is one bit time, however this can be adjusted by configuring the SWCLK frequency.

29.4.2. SWD protocol sequence

Each sequence consist of three phases:

- Packet request (8 bits) transmitted by the host
- Acknowledge response (3 bits) transmitted by the target
- Data transfer phase (33 bits) transmitted by the host or the target

Table 29-2 Packet request (8 bits)

Bit	Name	Description
0	Start	Must be “1”
1	APnDP	0: DP Access 1: AP Access
2	RnW	0: Write Request

		1: Read Request
4:3	A[3:2]	Address field of the DP or AP registers
5	Parity	Single bit parity of preceding bits
6	Stop	0
7	Park	Not driven by the host. Must be read as “1” by the target because of the pull-up

The packet request is always followed by the turnaround time (default 1 bit) where neither the host nor target drive the line.

Table 29-3 ACK response (3 bits)

Bit	Name	Description
[2:0]	ACK	001: FAULT 010: WAIT 100: OK

The ACK Response must be followed by a turnaround time only if it is a READ transaction or if a WAIT or FAULT acknowledge has been received.

Table 29-4 DATA transfer (33 bits)

Bit	Name	Description
[31:0]	WDATA or RDATA	Write or Read data
32	Parity	Single parity of the 32 data bits

The DATA transfer must be followed by a turnaround time only if it is a READ transaction.

29.4.3. SW-DP state machine (reset, idle states, ID code)

The State Machine of the SW-DP has an internal ID code which identifies the SW-DP. It follows the JEP-106 standard. This ID code is the default ARM one and is set to 0x0BB11477 (corresponding to Cortex®-M0).

29.4.4. DP and AP read/write accesses

- Read accesses to the DP are not posted: the target response can be immediate (if ACK = OK) or can be delayed (if ACK = WAIT).
- Read accesses to the AP are posted. This means that the result of the access is returned on the next transfer. If the next access to be done is NOT an AP access, then the DP-RDBUFF register must be read to obtain the result. The READOK flag of the DP-CTRL/STAT register is updated on every AP read access or RDBUFF read request to know if the AP read access was successful.
- The SW-DP implements a write buffer (for both DP or AP writes), that enables it toaccept a write operation even when other transactions are still outstanding. If the write buffer is full, the target acknowledge response is “WAIT”. With the exception of IDCODE read or CTRL/STAT read or ABORT write which are accepted even if the write buffer is full.
- Because of the asynchronous clock domains SWCLK and HCLK, two extra SWCLK cycles are needed after a write transaction (after the parity bit) to make the write effective internally. These cycles should be applied while driving the line low (IDLE state) This is particularly important when writing the CTRL/STAT for a power-up request. If the next transaction (requiring a power-up) occurs immediately, it will fail.

29.4.5. SW-DP registers

Access to these registers are initiated when APnDP = 0.

A[3:2]	R/W	CTRLSEL bit of SELECT register	Register	Notes
00	Read		IDCODE	The manufacturer code is set to the default ARM code for Cortex-M0: 0x0BB11477 (identifies the SW-DP)

00	Write		ABORT	
01	Read/Write	0	DP-CTRL/STAT	Purpose is to: – request a system or debug power-up – configure the transfer operation for AP accesses – control the pushed compare and pushed verify operations. – read some status flags (overrun, power-up acknowledges)
01	Read/Write	1	WIRE CONTROL	Purpose is to configure the physical serial port protocol (like the duration of the turnaround time)
10	Read		READ RESEND	Enables recovery of the read data from a corrupted debugger transfer, without repeating the original AP transfer.
10	Write		SELECT	The purpose is to select the current access port and the active 4-words register window
11	Read/Write		READ BUFFER	This read buffer is useful because AP accesses are posted (the result of a read AP request is available on the next AP transaction). This read buffer captures data from the AP, presented as the result of a previous read, without initiating a new transaction

29.4.6. SW-AP registers

Address	A[3:2] value	Description
0x0	00	Reserved
0x4	01	DP CTRL/STAT register. Used to: ■ Request a system or debug power-up ■ Configure the transfer operation for AP accesses ■ Control the pushed compare and pushed verify operations. ■ Read some status flags (overrun, power-up acknowledges)
0x8	10	DP SELECT register: Used to select the current access port and the active 4-words register window. ■ Bits 31:24: APSEL: select the current AP ■ Bits 23:8: reserved ■ Bits 7:4: APBANKSEL: select the active 4-words register window on the current AP ■ Bits 3:0: reserved
0xC	11	DP RDBUFF register: Used to allow the debugger to get the final result after a sequence of operations (without requesting new JTAG-DP operation)

29.5. Core debug

Core debug is accessed through the core debug registers. Debug access to these registers is by means of the debug access port. It consists of four registers:

Table 29-5 Core debug registers

Register	Description
DHCSR	32bit Debug halting control and status register
DCRSR	17bit Debug Core register selector register
DHCDR	32bit debug Core register Data register
DEMCR	32bit debug exception and monitor control register

These registers are not reset by a system reset. They are only reset by a power-on reset. Refer to the Cortex®-M0+ TRM for further details. To Halt on reset, it is necessary to:

- Enable the bit0 (VC_CORRESET) of the Debug and Exception Monitor Control Register
- Enable the bit0 (C_DEBUGEN) of the Debug Halting Control and Status Register

29.6. Break point unit (BPU)

The Cortex-M0+ BPU implementation provides four breakpoint registers. The BPU is a subset of the Flash Patch and Breakpoint (FPB) block available in ARMv7-M (Cortex-M3 & Cortex-M4).

29.6.1. BPU functionality

The processor breakpoints implement PC based breakpoint functionality.
Refer the ARMv6-M ARM and the ARM CoreSight Components Technical Reference Manual for more information about the BPU CoreSight identification registers, and their addresses and access types.

29.7. Data watchpoint (DWT)

The Cortex-M0 DWT implementation provides two watchpoint register sets

29.7.1. DWT functionality

The processor watchpoints implement both data address and PC based watchpoint functionality.

29.7.2. DWT program counter sample register

A processor that implements the data watchpoint unit also implements the ARMv6-M optional DWT Program Counter Sample Register (DWT_PCSR). This register permits a debugger to periodically sample the PC without halting the processor. This provides coarse grained profiling. See the ARMv6-M ARM for more information.
The Cortex-M0 DWT_PCSR records both instructions that pass their condition codes and those that fail.

29.8. MCU debug component (DBGMCU)

The MCU debug component helps the debugger provide support for:

- Low-power modes
- Clock control for timers, watchdog and I2C during a breakpoint

29.8.1. Debug support for low-power modes

To enter low-power mode, the instruction WFI or WFE must be executed. The MCU implements several low-power modes which can either deactivate the CPU clock or reduce the power of the CPU. The core does not allow FCLK or HCLK to be turned off during a debug session. As these are required for the debugger connection, during a debug, they must remain active. The MCU integrates special means to allow the user to debug software in low-power modes. For this, the debugger host must first set some debug configuration registers to change the low-power mode behavior:

- In Sleep mode: FCLK and HCLK are still active. Consequently, this mode does not impose any restrictions on the standard debug features.In stop mode: The DBG_STOP bit must be set in advance by the debugger.
- In Stop/Standby mode, the DBG_STOP bit must be previously set by the debugger. This enables the internal RC oscillator clock to feed FCLK and HCLK in Stop mode.

29.8.2. Debug support for times, watchdog and IIC

During a breakpoint, it is necessary to choose how the counter of timers and watchdog should behave:

- They can continue to count inside a breakpoint. This is usually required when a PWM is controlling a motor, for example.
- They can stop to count inside a breakpoint. This is required for watchdog purposes. For the I2C, the user can choose to block the SMBUS timeout during a breakpoint.

29.9. DBG register

29.9.1. DBG device ID code register (DBG_IDCODE)

Address offset: 0x00

Only supports 32-bit address access, read only.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Bit	Name	R/W	Reset Value	Function
31:0	Reserved	R		

29.9.2. Debug MCU configuration register (DBGMCU_CR)

This register configures the MCU low power mode in debug state.

This register is asynchronously reset by a power-on reset (not a system reset). It can be written by the debugger under system reset.

If the debugger host does not support this feature, it is still possible for the software user to write to these registers.

Address offset: 0x04

Reset value: 0x0000 0000 (will not be reset by system reset)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	DBG_ STOP	Res
														RW	

Bit	Name	R/W	Reset Value	Function
31:2	Reserved			
1	DBG_STOP	RW	0	Debug Stop mode 0: (FCLK = Off, HCLK = Off) In STOP mode, the clock controller disables HCLK and FCLK. When exiting from STOP mode, the clock configuration is identical to the one after RESET (CPU clocked by the HSI). Consequently, the software must reprogram the clock controller to enable the clock configuration. 1: (FCLK = on, HCLK = on). When entering STOP mode, HSI will not be turned off, and FCLK and HCLK are generated by I. When exiting STOP mode, if the clock control needs to be changed, the software needs to be reconfigured.
0	Reserved			

29.9.3. DBG APB freeze register 1 (DBG_APB_FZ1)

This register is used to configure the clock of timer, RTC, IWDG, WWDG under debug. This register is asynchronously reset by a power-on reset (not a system reset). It can be written by the debugger under system reset.

Address offset: 0x08

Power on Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DBG_LPTIM_STOP	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
RW															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	DBG_IWDG_STOP	DBG_WWDG_STOP	DBG_RTC_STOP	Res	Res	Res	Res	Res	Res	Res	Res	DBG_TIM3_STOP	Res
			RW	RW	RW									RW	

Bit	Name	R/W	Reset Value	Function
31	DBG_LPTIM_STOP	RW	0	When the CPU is stopped, the counter clock control bit of the L PTIM 0: enable 1: Disable
30:13	Reserved			
12	DBG_IWDG_STOP	RW	0	When the CPU is stopped, the clock control bit of the IWDG counter 0: enable 1: Disable
1 1	DBG_WWDG_STOP	RW	0	When the CPU is stopped, the clock control bit of the WWDG counter 0: enable 1: Disable
1 0	DBG_RTC_STOP	RW	0	When the CPU is stopped, the clock control bit of the RTC counter 0: enable 1: Disable
9:2	Reserved			
1	DBG_TIM3_STOP	RW	0	TIM 3 counter when the CPU is stopped 0: enable 1: Disable
0	Reserve			

29.9.4. DBG APB freeze register 2 (DBG_APB_FZ2)

This register is used to configure the clock control of timer under debug. This register is asynchronously reset by a power-on reset (not a system reset). It can be written by the debugger under system reset.

Address offset: 0x0C

Power on Reset value: 0x0000 0000

Only supports 32-bit address access, read only.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	DBG_TIM17_STOP	DBG_TIM16_STOP	Res
													RW	RW	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DBG_TIM14_STOP	Res	Res	Res	DBG_TIM1_STOP	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
RW				RW											

Bit	Name	R/W	Reset Value	Function
31:19	Reserved			
18	DBG_TIM17_STOP			When the CPU is stopped, the clock control bit of the TIM17 counter

				0: Enable 1: Disable
17	DBG_TIM16_STOP			When the CPU is stopped, the clock control bit of the TIM16 counter 0: Enable 1: Disable
16	Reserved			
15	DBG_TIM14_STOP			When the CPU is stopped, the clock control bit of the TIM14 counter 0: Enable 1: Disable
14:12	Reserved			
11	DBG_TIM1_STOP			When the CPU is stopped, the clock control bit of the TIM1 counter 0: Enable 1: Disable
10:0	Reserved			

29.9.5. DBG register map

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x000	DBG_ID-CODE	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x004	DBG_CR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBG_STO	Res.
	Reset value																															0	
0x008	DBG_APB_FZ1	DBG_LPTIM_STO	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBG_IWDG_STO	DBG_WWDG_ST	DBG_RTC_STOP	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBG_TIM3_STOP	Res.
	Reset value	0																			0	0	0									0	
0x00C	DBG_APB_FZ2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBG_TIM17_ST	DBG_TIM16_ST	Res.	DBG_TIM14_ST	Res.	Res.	Res.	DBG_TIM1_STO	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value														0	0		0				0											

30. Version history

Version	date	update record	Author
V1.0	2021.10.20	First edition	LSQ
V1.1	2022.04.13	1. Table 4-5, modify parameters 2. Modify the description of the status register (USART_SR) 3. Figure 15-1, Modified Comparator Architecture Block Diagram	LSQ
V1.2	2022.06.24	English version	WYK/HJY/LJY/GHY