**18.3 Функциональное описание TIM2.**

**18.3.1 Блок временной базы.**

Главный блок программируемого таймера представляет собой 16-ти битный или 32-битный счетчик со связанным с ним авто-перезагружаемым регистром. Счетчик может считать вверх, вниз, или в обе стороны. Частота счета может быть поделена прескалером.

Счетчик, авто-перезагружаемый регистр и регистр прескалера могут быть записаны или прочитаны программой. Это справедливо даже если счетчик уже запущен.

Блок временной базы включает:

* Регистр счетчика (TIMx\_CNT)
* Регистр прескалера (TIMx\_PSC)
* Авто-перезагружаемый регистр (TIMx\_ARR)

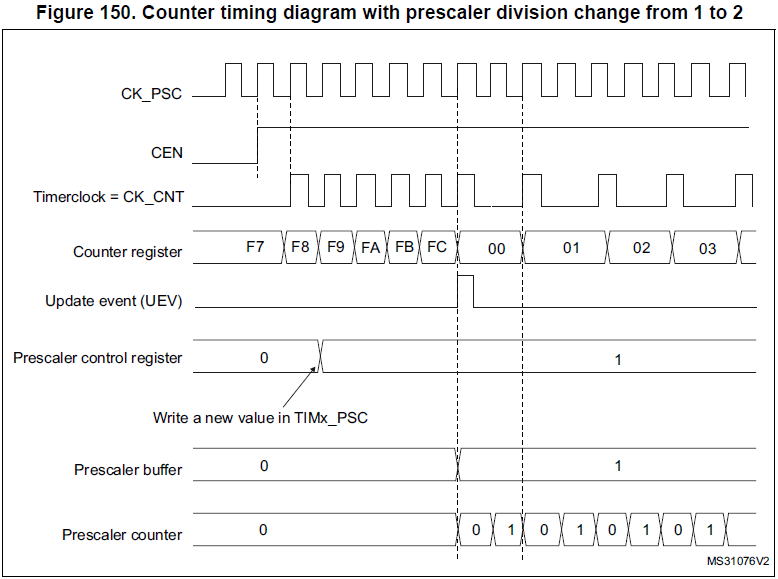
Авто-перезагружаемый регистр является предзагружаемым. Запись или чтение из него производится фактически через регистр предзагрузки. Содержимое регистра предзагрузки выгружается в теневой регистр постоянно, или на каждом событии обновления (UEV), в зависимости от бита разрешения предзагрузки ARPE в регистре TIMx\_CR1. Событие обновления происходит, когда счетчик достигает переполнения (или опустошения, если считает вниз), и если UDIS бит равен 0 в регистре TIMx\_CR1. Оно также может быть вызвано программно. Генерация события обновления описана в деталях для каждой конфигурации.

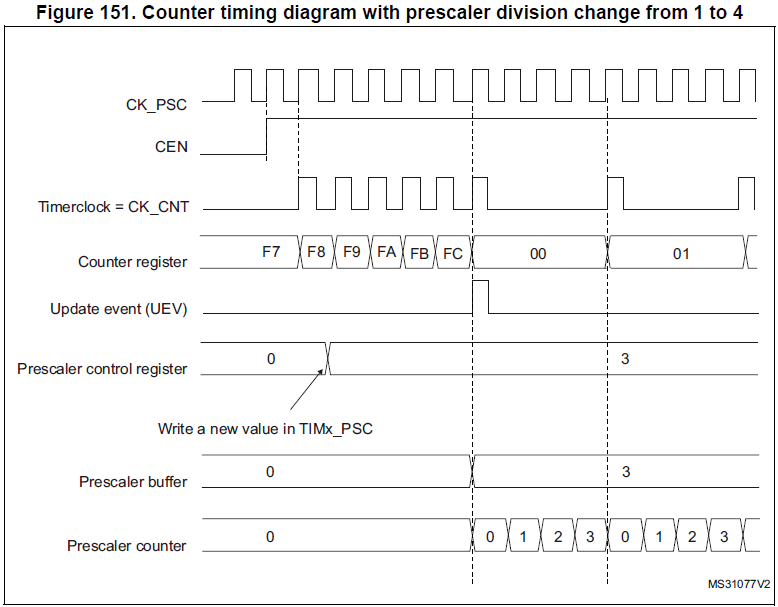
Счетчик тактируется выходом прескалера CK\_CNT, который разрешен только когда установлен бит CEN в регистре TIMx\_CR1. Заметьте, что фактический сигнал разрешения счета устанавливается через один такт после установки CEN.

**Описание прескалера.**

Прескалер может делить тактовую частоту счета на любой коэффициент между 1 и 65536. Он построен на основе 16-битного счетчика, управляемого через 16/32-битный регистр TIMx\_PSC. Коэффициент деления может быть изменен на лету, так как его регистр управления буферизован. Новый коэффициент деления вступит в силу на следующем событии обновления.

Рисунок 150 и 151 дают некоторые примеры поведения счетчика при изменении прескалера на лету.





**18.3.2 Режимы счета.**

**Режим счета вверх.**

В данном режиме счетчик считает от 0 до авто-перезагружаемого значения (содержимого регистра TIMx\_ARR), затем перезапускается с 0 и генерирует событие переполнения счетчика.

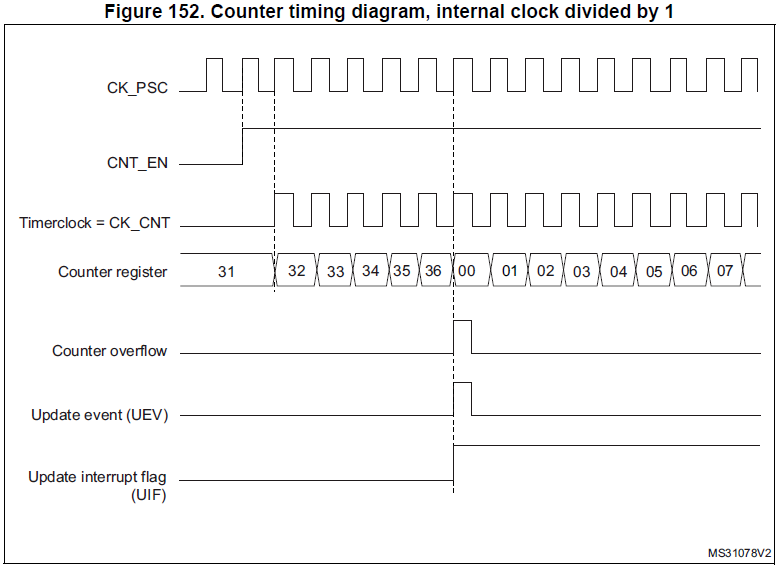
Событие переполнения может быть сгенерировано на каждом переполнении счетчика, или путем установки бита UG в регистре TIMx\_EGR (программно, или используя slave mode controller).

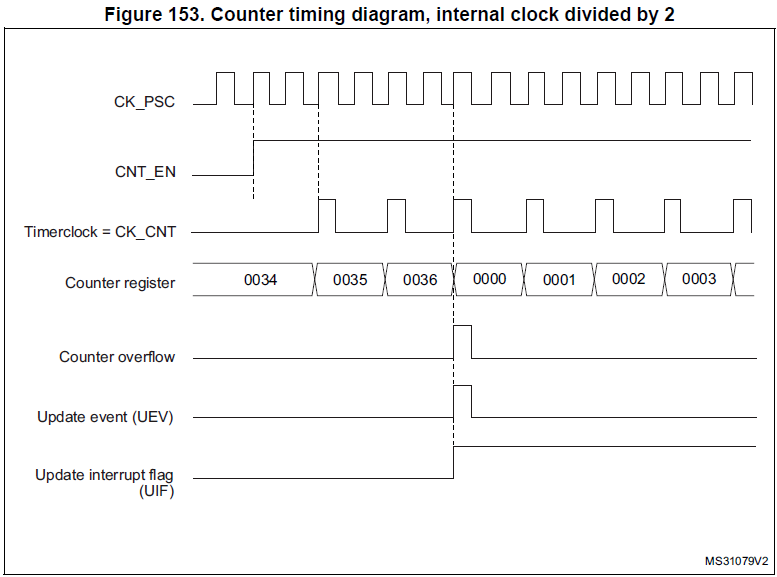
UEV событие может быть запрещено программно, путем установки бита UDIS в регистре TIMx\_CR1. Это может понадобиться, чтобы избежать обновления теневых регистров во время записи новых значений в регистры предзагрузки. При этом события обновления не возникают до тех пор, пока бит UDIS остается равен 0. Однако, счетчик вновь начнет считать с нуля, ровно как и его прескалер (хотя скорость прескалера не будет изменяться). Кроме этого, если URS бит (update request selection) в регистре TIMx\_CR1 установлен, установка UG бита генерирует событие обновления UEV без установки флага UIF (и таким образом, не посылая DMA-запрос). Это может понадобиться, чтобы избежать генерацию обоих типов прерывания – обновления и захвата при очистке счетчика по событию захвата.

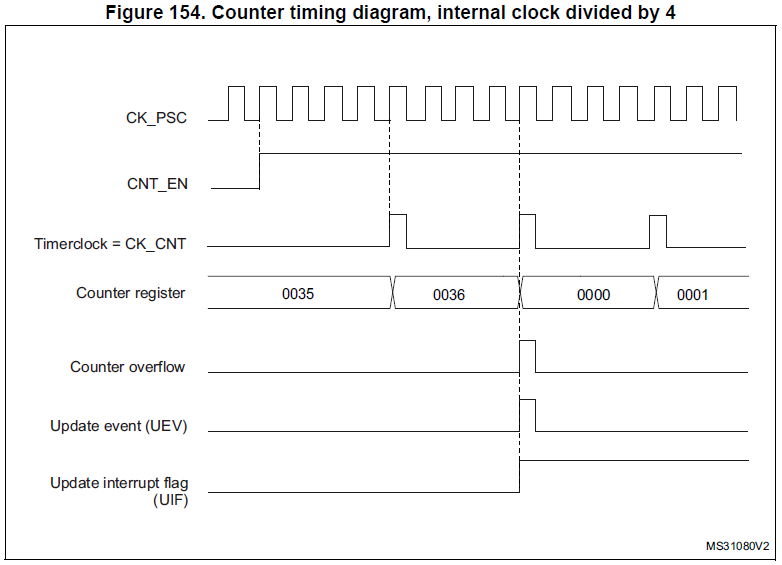
Когда происходит событие обновления, все регистры обновляются и флаг обновления (UIF бит в регистре TIMx\_SR) устанавливается (в зависимости от URS бита):

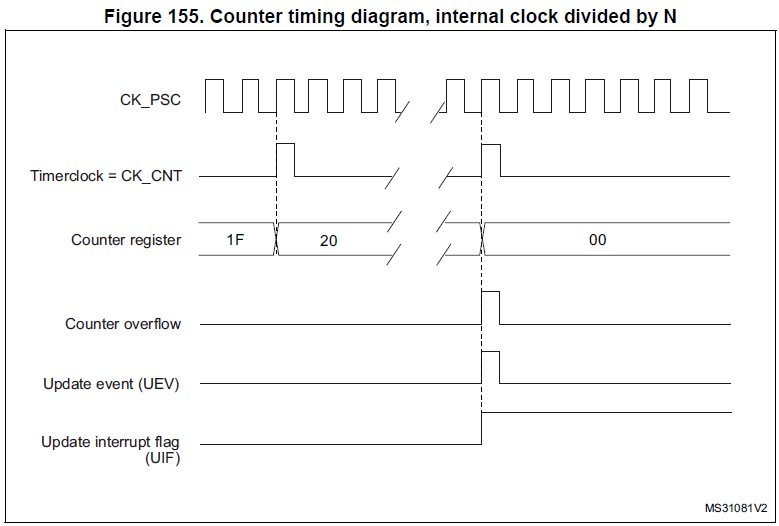
* Буфер прескалера перезагружается значением предзагрузки (содержимым регистра TIMx\_PSC).
* Авто-перезагружаемый теневой регистр обновляется значеним предзагрузки (TIMx\_ARR).

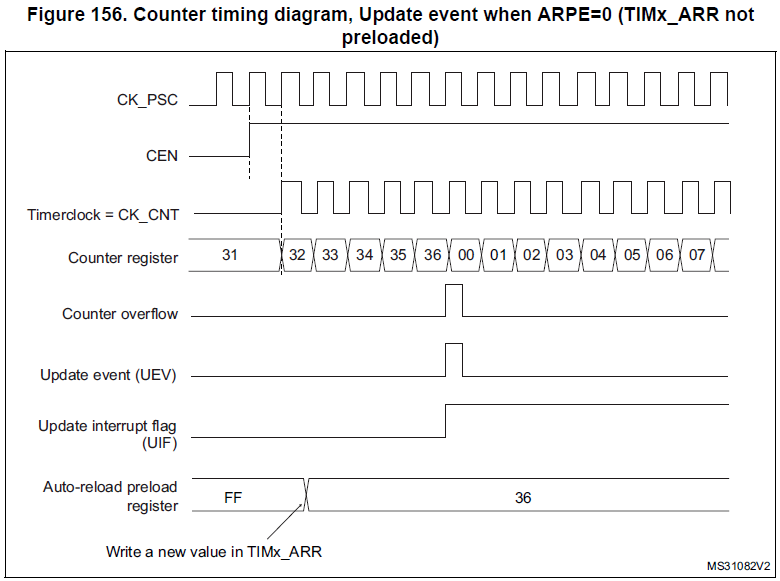
Следующие рисунки показывают некоторые примеры поведения счетчика для различных тактовых частот, при TIMx\_ARR=0x36.

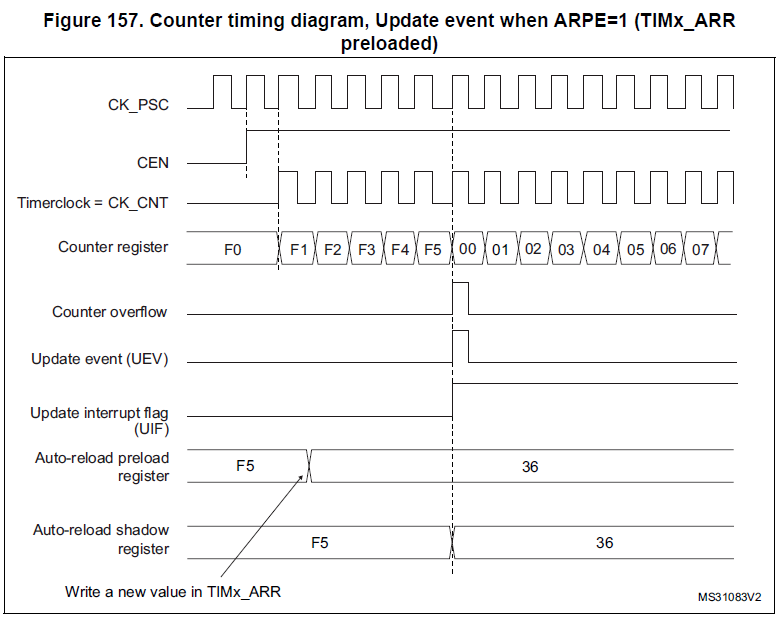












**18.3.8 Режим output compare.**

Эта функция используется для управления выходным сигналом или индикации факта истечения некоторого периода времени.

При обнаружении факта совпадения между регистром захвата/сравнения и счетчиком, функция output compare:

* Устанавливает соответствующий выходной вывод в значение, определяемое установленным режимом output compare (OCxM биты в регистре TIMx\_CCMRx) и установленной полярностью выхода (CCxP бит в регистре TIMx\_CCER). Выходной вывод может сохранить свой уровень (OCxM=000), установиться в активный уровень (OCxM=001), установиться в неактивный уровень (OCxM=010), или может переключиться (OCxM=011) при совпадении.
* Устанавливает флаг в регистре статуса прерывания, если настроена соответствующая маска прерывания (CCxIE бит в регистре TIMx\_DIER).
* Посылает DMA запрос, если соответствующий бит разрешения установлен (CCxDE бит в регистре TIMx\_DIER, CCDS бит в регистре TIMx\_CR2 для выбора запроса DMA).

TIMx\_CCRx регистры могут быть запрограммированы посредством предзагрузки или без нее, в зависимости от бита OCxPE в регистре TIMx\_CCMRx.

В режиме output compare, событие обновления UEV не влияет на выходы OCxREF и OCx (NB: это значит, что выход OCx будет изменяться только по событию совпадения, а следовательно, что данный режим не позволит выдать ШИМ-сигнал). Output compare режим может быть также использован для выдачи однократного импульса (режим One-pulse).

**18.3.9 Режим ШИМ.**

Режим ШИМ позволяет вам выдать сигнал с частотой, определяемой TIMx\_ARR регистром и рабочим циклом, определяемым значением регистра TIMx\_CCRx.

Режим ШИМ может быть независимо выбран на каждом канале (по одному ШИМ на OCx выход) путем записи 110 (PWM режим 1) или 111 (PWM режим 2) в биты OCxM регистра TIMx\_CCMRx. Вы должны разрешить соответствующий регистр предзагрузки путем установки OCxPE бита в регистре TIMx\_CCMRx а также разрешить атоперезагрузку для этого регистра (в режиме счета вверх или center-aligned режиме) путем установки бита ARPE в регистре TIMx\_CR1.

Так как регистр прдзагрузки выгружается в теневой регистр только по событию обновления, то перед запуском счетчика вам необходимо инициализировать все регистры, установив бит UG в регистре TIMx\_EGR.

Полярность OCx программируется битом CCxP в регистре TIMx\_CCER. Она может соответствовать высокому или низкому активному уровню. OCx выход разрешается битом CCxE в регистре TIMx\_CCER. Обратитесь к описанию регистра TIMx\_CCERx за деталями.

В режиме ШИМ (1 или 2) TIMx\_CNT и TIMx\_CCRx всегда сравниваются между собой для определения факта истины TIMx\_CCRx ≤ TIMx\_CNT или TIMx\_CNT ≤ TIMx\_CCRx (в зависимости от направления счета). Однако, чтобы соответствовать функциональности OCREF\_CLR (OCREF может быть очищен по внешнему событию сигналом ETR до следующего периода ШИМ) OCREF сигнал устанавливается в активный уровень только когда:

* When the result of the comparison or (NB: не дописано...?)
* Когда режим output compare (OCxM биты в регистре TIMx\_CCMRx) переключается из «замороженной» конфигурации (нет сравнения, OCxM=000) в один из ШИМ режимов (OCxM=110 или 111).

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

Таймер может генерировать ШИМ в edge-aligned режиме или center-aligned режиме в зависимости от битов CMS в регистре TIMx\_CR1.

**Edge-aligned режим ШИМ.**

Upcounting конфигурация.

Режим счета вверх активен, когда бит DIR в регистре TIMx\_CR1 равен «0».

В следующем примере мы рассмотрим PWM режим 1. Опорный сигнал OCxREF равен «1» до тех пор пока TIMx\_CNT<TIMx\_CCRx, и «0» в противном случае. Если значение сравнения в TIMx\_CCRx больше чем auto-reload значение (в TIMx\_ARR), то OCxREF удерживается в «1» всегда. Если значение сравнения равно нулю, то OCxREF удерживается в «0» всегда. Рисунок 179 показывает пример выдачи ШИМ, при TIMx\_ARR=8.

