

15 ТАЙМЕРЫ

Процессор имеет три идентичных 32-разрядных таймера общего назначения, таймер ядра и сторожевой таймер.

Для каждого из таймеров общего назначения может быть индивидуально задан один из трёх режимов работы:

- режим широтно-импульсной модуляции (PWM_OUT),
- режим измерения периода и длительности импульса (WDTH_CAP),
- режим счётчика внешних воздействий (EXT_CLK).

Таймер ядра генерирует периодические прерывания, используемые в задачах синхронизации системы.

Сторожевой таймер может использоваться для реализации функции программного сторожа. Применение программного сторожа позволяет повысить устойчивость работы системы при помощи генерации событий ядра процессора, если значение таймера достигает нуля до того, как он будет сброшен программно.

Таймеры общего назначения

Каждый таймер общего назначения имеет один выделенный двунаправленный вывод, TMR_x. Этот вывод является входным в режиме PWM_OUT и выходным в режимах WDTH_CAP и EXT_CLK. Для обеспечения работы в этих режимах, каждый таймер имеет четыре регистра. Регистры счётчика (TIMER_x_COUNTER), периода (TIMER_x_PERIOD) и длительности импульса (TIMER_x_WIDTH) имеют разрядность 32 бита, что повышает диапазон значений и точность таймера. Устройство таймера показано на рис. 15-1.

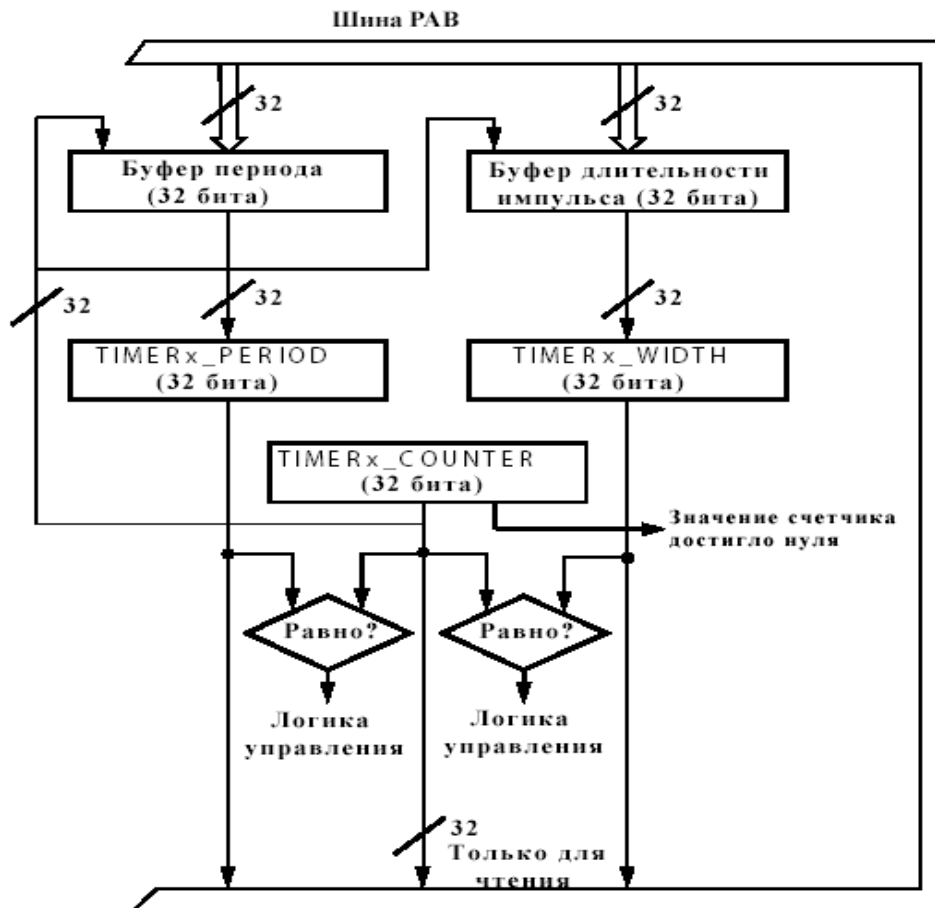


Рис. 15-1. Блок-схема таймера

Каждый таймер общего назначения включает следующие регистры:

- регистр конфигурации (TIMERx_CONFIG);
- регистр счётчика (TIMERx_COUNTER);
- регистр периода (TIMERx_PERIOD);
- регистр длительности импульса (TIMERx_WIDTH).

Когда таймер работает от внутреннего тактового сигнала, источником сигнала является тактовый сигнал периферийных устройств процессора (SCLK). Предположим, что частота сигнала SCLK составляет 133 МГц, тогда максимальный период счётчика таймера равен $((2^{32} - 1) / 133 \text{ МГц}) = 32.2 \text{ с}$.

Регистр включения таймеров (TIMER_ENABLE) может использоваться для одновременного включения всех трёх таймеров. Этот регистр содержит три бита управления, имеющие тип “запись-1-для-установки”, – по одному для каждого таймера. Регистр выключения таймеров (TIMER_DISABLE) содержит три бита управления, имеющие тип “запись-1-для-сброса”, позволяющие одновременно или независимо выключать таймеры. Для получения информации о состоянии таймеров можно выполнить чтение любого из этих регистров. Возвращаемое при чтении значение 1 означает, что соответствующий таймер включён. Таймер начинает работу через три такта SCLK после установки бита TIMENx.

15 Таймеры

Регистр состояния таймеров (TIMER_STATUS) содержит биты фиксации прерывания (TIMILx) и индикации ошибки/переполнения (TOVF_ERRx) для каждого таймера. Эти “залипающие” биты устанавливаются аппаратно и могут опрашиваться программой. Сброс этих битов должен выполняться программно путём записи единицы в соответствующий бит.

Для разрешения генерации прерываний таймера следует установить бит IRQ_ENA в регистре конфигурации таймера (TIMERx_CONFIG) и снять маскирование прерывания таймера, установив соответствующие биты в регистрах IMASK и SIC_IMASK. Когда бит IRQ_ENA сброшен, таймер не устанавливает биты фиксации прерывания (TIMILx). Для опроса состояния битов TIMILx без генерации прерываний таймера, можно установить бит IRQ_ENA, оставив бит прерывания маскированным.

Когда генерация прерывания разрешена, необходимо, чтобы в программе обслуживания прерывания бит TIMILx сбрасывался до выполнения команды RTI. Это нужно для того, чтобы исключить повторное срабатывание прерывания. В режиме счётчика внешних воздействий (EXT_CLK) бит TIMILx должен быть сброшен в самом начале программы обслуживания прерывания, чтобы не пропустить изменения состояния таймера. Для разрешения генерации прерывания таймером нужно установить бит IRQ_ENA в соответствующем регистре конфигурации таймера (TIMERx_CONFIG).

Регистры таймеров

Функции таймеров общего назначения обеспечиваются периферийным модулем, состоящим из трёх идентичных таймеров.

Каждый таймер имеет четыре регистра:

- TIMERx_CONFIG[15:0] – регистр конфигурации таймера;
- TIMERx_WIDTH[31:0] – регистр длительности импульса таймера;
- TIMERx_COUNTER[31:0] – регистр счётчика таймера;
- TIMERx_PERIOD[31:0] – регистр периода таймера;

Следующие три регистра совместно используются тремя таймерами:

- TIMER_ENABLE[15:0] – регистр включения таймеров;
- TIMER_DISABLE[15:0] – регистр выключения таймеров;
- TIMER_STATUS[15:0] – регистр состояния таймеров;

Разрядность операций обращения к перечисленным регистрам должна строго соблюдаться. 32-разрядные обращения к регистру конфигурации таймера или 16-разрядные обращения к регистрам длительности импульса, периода или счётчика таймера вызывают ошибку обращения к регистрам, отображённым в карте памяти. Обращение к регистрам состояния таймеров, включения таймеров и выключения таймеров возможны при помощи 16- и 32-разрядных операций. При выполнении 32-разрядной операции чтения этих регистров старшая часть возвращаемого слова содержит нули.

Регистр включения таймеров (TIMER_ENABLE)

Регистр включения таймеров позволяет выполнять одновременное включение всех трёх таймеров; при этом достигается полная синхронность их работы. Каждому таймеру в регистре соответствует бит управления типа W1S. Запись значения 1 разрешает работу соответствующего таймера; запись значения 0 не влияет на работу таймера. Биты могут устанавливаться по отдельности или в любой комбинации. При чтении регистра включения таймеров отображается состояние соответствующего таймера. Значение 1 означает, что таймер включён. В позициях незадействованных битов при чтении возвращаются нули.

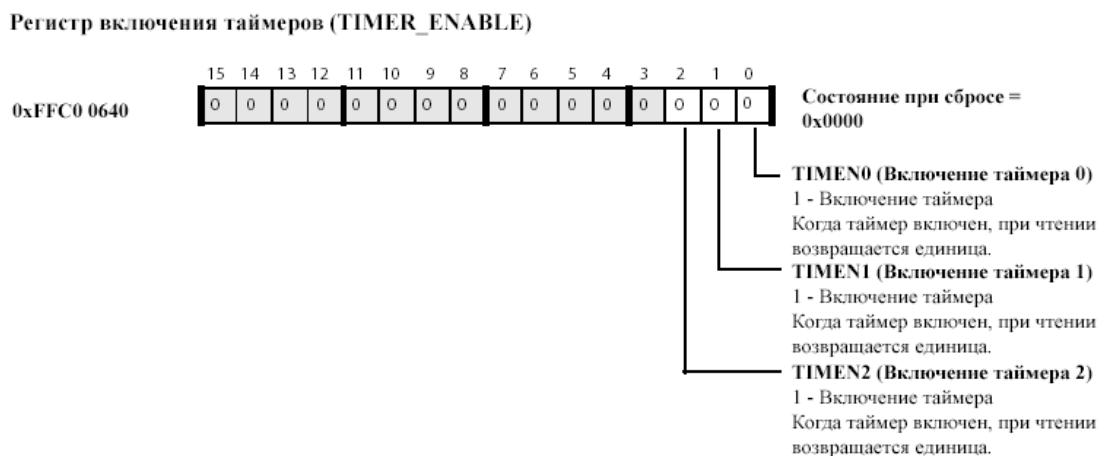


Рис. 15-2. Регистр включения таймеров

Регистр выключения таймеров (TIMER_DISABLE)

Регистр выключения таймеров позволяет выполнять одновременное выключение всех трёх таймеров. Каждому таймеру в регистре соответствует бит управления типа W1C. Запись значения 1 выключает соответствующий таймер; запись значения 0 не влияет на работу таймера. Биты могут устанавливаться по отдельности или в любой комбинации. При чтении регистра выключения таймеров возвращается значение, идентичное значению, возвращаемому при чтении регистра включения таймеров. Значение 1 означает, что таймер включён. В позициях незадействованных битов при чтении возвращаются нули.

Регистр выключения таймеров (TIMER_DISABLE)

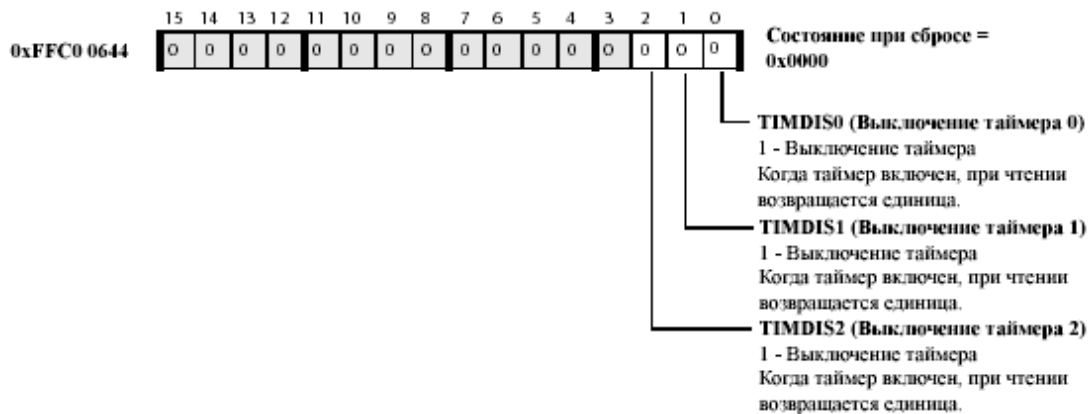


Рис. 15-3. Регистр выключения таймеров

В режиме PWM_OUT запись единицы в бит регистра TIMER_DISABLE не приводит к мгновенной остановке работы соответствующего таймера. Вместо этого таймер продолжает работу и останавливается только в конце текущего периода (если PERIOD_CNT = 1) или импульса (если PERIOD_CNT = 0). При необходимости процессор может принудительно вызвать мгновенную остановку работы таймера в режиме PWM_OUT записью единицы в соответствующий бит регистра TIMER_DISABLE и последующей записью единицы в соответствующий бит TRUNx регистра TIMER_STATUS (См. раздел “Остановка таймера в режиме PWM_OUT”).

В режимах WDTN_CAP и EXT_CLK запись единицы в бит регистра TIMER_DISABLE вызывает незамедлительную остановку работы соответствующего таймера.

Регистр состояния таймеров (TIMER_STATUS)

Регистр состояния таймеров отображает состояние таймеров; он даёт возможность получения информации о состоянии всех трёх таймеров при помощи одной команды чтения. Биты состояния являются “залипающими” и имеют тип W1C. Биты TRUNx могут сбрасываться без воздействия пользователя, при остановке таймера в конце периода в режиме PWM_OUT. При чтении регистра состояния в позициях незадействованных или зарезервированных битов возвращаются нули.

Каждый таймер генерирует отдельный запрос прерывания, который разрешается соответствующим битом IRQ_ENA в регистре TIMERx_CONFIG. В регистре состояния таймеров (TIMER_STATUS), совместно используемом тремя таймерами, фиксируется каждое из этих прерываний; таким образом, пользователь может определить источник прерывания без задания отдельного сигнала прерывания (например, в случае, когда всем трём прерываниям таймера назначен одинаковый приоритет). Биты прерываний являются залипающими и

15 Таймеры

должны сбрасываться программой обслуживания прерывания, чтобы предотвратить повторное срабатывание прерывания.

При установке бита `IRQ_ENA` регистра конфигурации таймера биты `TIMILx` используются для сигнализирования о запросе прерывания. Если возникает ошибка или условие прерывания и бит `IRQ_ENA` установлен, устанавливается бит `TIMILx` и генерируется прерывание ядра. Это прерывание может маскироваться контроллером прерываний системы. Если возникает ошибка или условие прерывания и бит `IRQ_ENA` сброшен, бит `TIMILx` не устанавливается и прерывание ядра не генерируется. Если на момент записи нуля в бит `IRQ_EN` бит `TIMILx` установлен, он сохраняет своё состояние и прерывание остаётся активным.

Значения битов `TRUNx`, получаемые при чтении, отображают состояние таймера во всех режимах: если бит `TRUNx` установлен – таймер работает, если бит `TRUNx` сброшен – таймер остановлен. При чтении битов `TIMENx` и `TIMDISx` в регистрах `TIMER_ENABLE` и `TIMER_DISABLE` возвращается информация о том, включён ли таймер; бит `TRUNx` указывает, работает ли таймер. В режимах `WDTH_CAP` и `EXT_CLK` при чтении битов `TIMENx` и `TRUNx` всегда возвращаются одинаковые значения.

Операция `W1C` в регистр `TIMER_DISABLE` выключает соответствующий таймер во всех режимах. В режиме `PWM_OUT` выключаемый таймер продолжает работать до завершения текущего периода (`PERIOD_CNT = 1`) или до окончания импульса (`PERIOD_CNT = 0`). В течение последнего периода при чтении бита `TIMENx` возвращается 0, а при чтении `TRUNx` – 1 (см. рис. 15-10). Только в этом состоянии бит `TRUNx` имеет тип `W1C`. Если в течение периода, в котором выполняется выключения таймера, выполнить запись единицы в бит `TRUNx`, то этот бит будет сброшен и таймер незамедлительно остановится, не дожидаясь завершения текущего цикла счетчика таймера.

Запись в биты `TRUNx` при работе в других режимах или при выключенном таймере не изменяет их содержимого. В режиме `PWM_OUT` запись единицы в бит `TRUNx` не влияет на работу таймера, если перед этим не выполнялась операция его выключения.

Регистр состояния таймера (TIMER_STATUS)

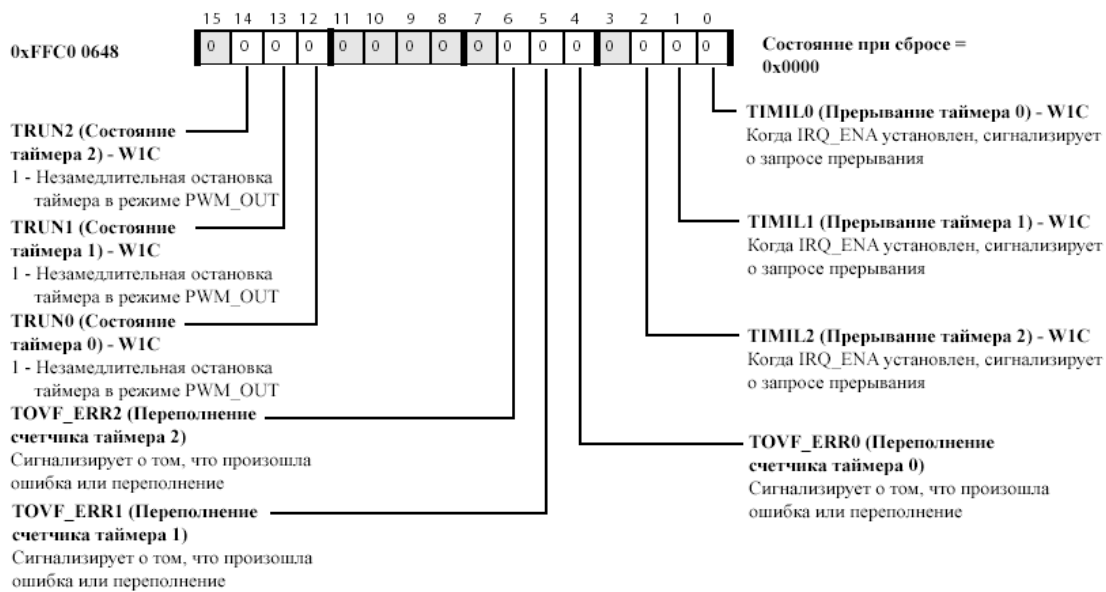


Рис. 15-4. Регистр состояния таймеров

Регистры конфигурации таймеров (TIMERx_CONFIG)

Рабочий режим каждого из таймеров задается в регистре конфигурации таймера. Запись в регистр `TIMERx_CONFIG` возможна только, когда работа таймера остановлена. Перед попыткой программного изменения регистра `TIMERx_CONFIG` после выключения таймера в режиме `PWM_OUT`, необходимо убедиться, что таймер остановлен, проверив состояние бита `TRUNx` в регистре `TIMERx_STATUS`. Чтение регистра `TIMERx_CONFIG` возможно в любой момент времени. Поле `ERR_TYP` доступно только для чтения. Оно сбрасывается при сбросе процессора и при включении таймера. Каждый раз, когда устанавливается бит `TOVF_ERR`, в поле `ERR_TYP[1:0]` записывается код, определяющий тип обнаруженной ошибки. Значение поля сохраняется до тех пор, пока не возникнет следующая ошибка или не будет выполнена запись в бит включения таймера.

Обзор условий ошибок см. в таблице 15-1. Регистр `TIMERx_CONFIG` также управляет поведением вывода `TMRx`; при сбросе бита `OUT_DIS` в режиме `PWM_OUT` (`TMODE = 01`) разрешается выходной сигнал на этом выводе.

Регистры счетчиков таймеров (TIMERx_COUNTER)

Эти регистры, доступные только для чтения, сохраняют свое состояние при выключении таймера. При включении таймера, регистр счетчика таймера аппаратно инициализируется в зависимости от конфигурации и режима. Чтение регистра счетчика таймера возможно в любой момент времени (независимо от того, работает ли таймер или остановлен), при этом возвращается когерентное

15 Таймеры

32-разрядное значение. В зависимости от рабочего режима тактовый сигнал счетчика может поступать от одного из четырех источников: SCLK, вывода TMRx, вывода программируемого флага PF1 или сигнала тактовой синхронизации параллельного порта PPI_CLK.

Когда внешний эмулятор выполняет доступ к ядру процессора, выполнение программы останавливается. По умолчанию работа счётчика TIMERx_COUNTER также останавливается при эмуляции для сохранения синхронизации с программой. Когда счетчик остановлен, он не инкрементируется – в режиме PWM_OUT происходит “растягивание” сигнала на выводе TMRx, в режиме EXT_CLK возможен пропуск входных воздействий на выводе TMRx. Все остальные функции, такие как чтение и запись регистров, ранее сгенерированные (и не сброшенные) прерывания и загрузка регистров TIMERx_PERIOD и TIMERx_WIDTH в режиме WDTN_CAP, при остановке таймера в режиме эмуляции остаются активными.

В некоторых приложениях может потребоваться, чтобы таймер продолжал работу асинхронно с ядром процессора, остановленным при эмуляции. Для этого следует установить бит EMU_RUN в регистре TIMERx_CONFIG.

Регистры счетчиков таймеров (TIMERx_COUNTER)

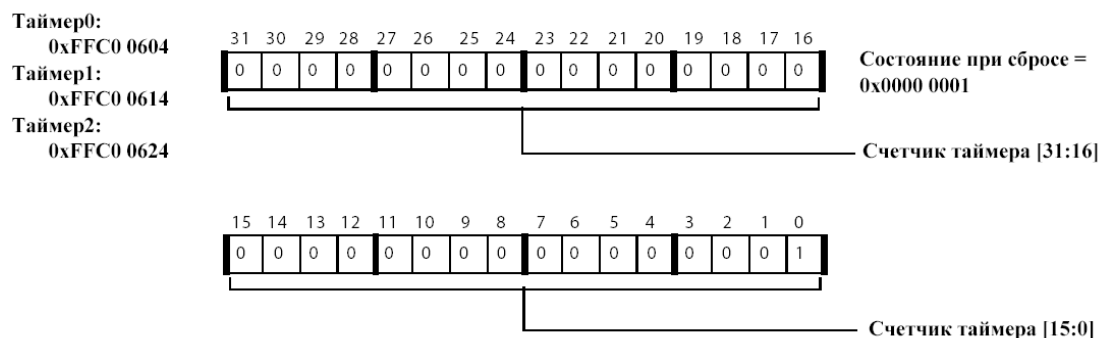


Рис. 15-6. Регистры счетчиков таймеров

Регистры периода таймеров (TIMERx_PERIOD) и регистры длительности импульса таймеров (TIMERx_WIDTH).

i Когда таймер включён и работает, при выполнении программой записи новых значений в регистр периода и регистр длительности импульса таймера, записываемое значение буферизуется и содержимое регистров не обновляется до окончания текущего периода (момента, когда значение регистра счетчика станет равным значению регистра периода).

Использование регистров периода и длительности импульса зависит от режима работы таймера:

15 Таймеры

- В режиме широтно-импульсной модуляции (PWM_OUT) значения и регистра периода, и регистра длительности импульса могут обновляться “на лету”, так как они изменяются одновременно.
- В режиме измерения длительности импульса и периода (WDTH_CAP) захват значений буферов периода и длительности импульсов происходит в определенный момент времени. Затем выполняется одновременное обновление регистров периода и длительности импульса содержимым соответствующих буферов. В данном режиме оба регистра доступны только для чтения.
- В режиме счётчика внешних событий (EXT_CLK) возможна запись в регистр периода таймера и его обновление “на лету”. Регистр длительности импульса не используется.

Если в регистр периода или регистр длительности импульса таймера не записывается новое значение, повторно используется значение, которое использовалось в предыдущем периоде. Запись в 32-разрядные регистры периода и длительности импульса является элементарной операцией; невозможна отдельная запись старшего слова без выполнения записи младшего слова.

Значения, записываемые в регистры периода и регистры длительности импульса таймеров, всегда помещаются в буферные регистры. При чтении регистров периода или длительности импульса всегда возвращается текущее активное значение периода или длительности импульса. Записанные значения не будут возвращаться при чтении до тех пор, пока они не станут активными. Когда таймер включён, записанные значения не станут активными, пока значения регистров периода и длительности импульса не будут обновлены содержимым соответствующих буферов в конце текущего периода (см. рис. 15-1).

Когда таймер выключен, значения, записываемые в буферные регистры, незамедлительно копируются в регистры периода и длительности импульса таймера и становятся доступны для использования в первом периоде работы таймера. Например, для того, чтобы изменить значения регистров периода и длительности импульса таймера таким образом, чтобы в первых трех периодах после включения таймера использовались различные настройки, следует воспользоваться следующей процедурой:

1. Задайте первый набор значений регистров.
2. Включите таймер.
3. Сразу же задайте второй набор значений регистров.
4. Дождитесь первого прерывания таймера.
5. Задайте третий набор значений регистров.

Каждое новое значение задается при получении прерывания таймера.



В режиме PWM_OUT с очень малыми значениями периода (менее 10) может не хватать времени на обновление и регистра периода, и регистра длительности импульса таймера содержимым буферных регистров. В этом случае в следующем периоде может использоваться одно новое и одно старое значение. Для предотвращения ошибок, возникающих из-за того, что значение длительности импульса больше или равно значению

15 Таймеры

периода, необходимо при уменьшении значений выполнять запись в регистр длительности импульса в первую очередь, а при увеличении значений – в регистр периода.

Регистры периода таймеров (TIMERx_PERIOD)

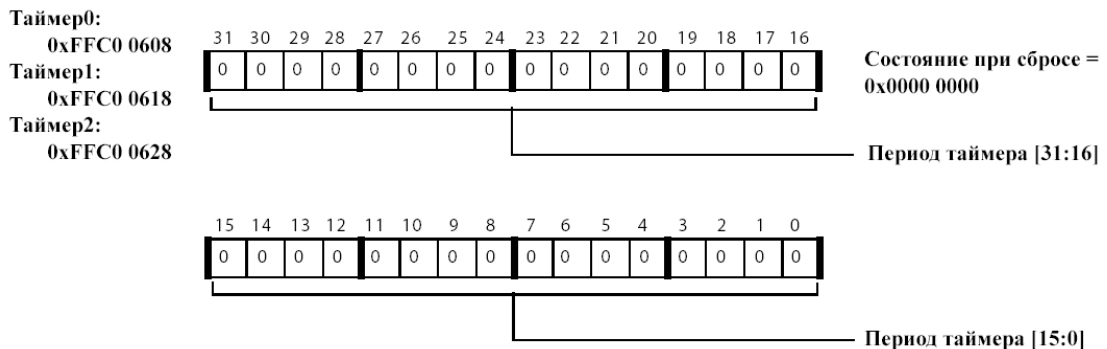


Рис. 15-7. Регистры периода таймеров

Регистры длительности импульса таймеров (TIMERx_WIDTH)

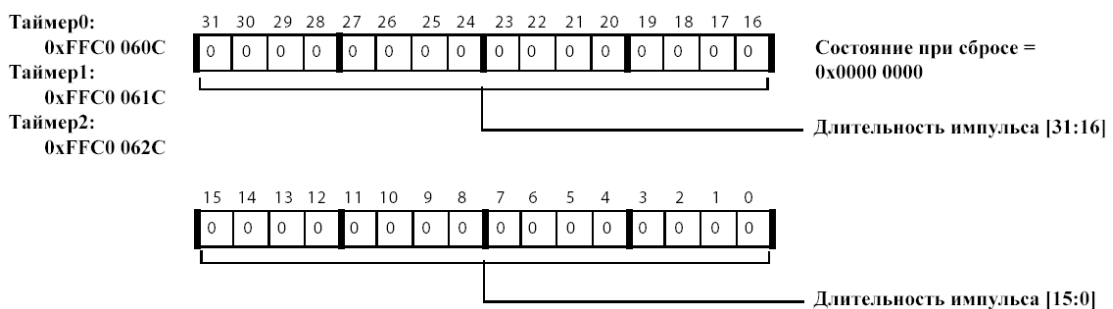


Рис. 15-8. Регистры длительности импульса таймеров.

Использование таймера

Для включения отдельного таймера следует установить соответствующий бит `TIMEN` в регистре `TIMER_ENABLE`. Для выключения отдельного таймера следует установить соответствующий бит `TIMDIS` в регистре `TIMER_DISABLE`. Для одновременного включения всех трех таймеров необходимо установить все три бита `TIMEN` в регистре `TIMER_ENABLE`.

Перед включением таймера всегда следует выполнять программирование соответствующего регистра конфигурации таймера (`TIMERx_CONFIG`). Этот регистр определяет рабочий режим таймера, полярность вывода `TMRx` и необходимость генерации прерываний таймера. Не следует изменять рабочий режим во время работы таймера.

15 Таймеры

Примеры временных диаграмм включения и выключения таймера показаны на рис. 15-9, 15-10 и 15-11.

Пример временной диаграммы включения таймера (режим PWM_OUT, PERIOD_CNT = 1)

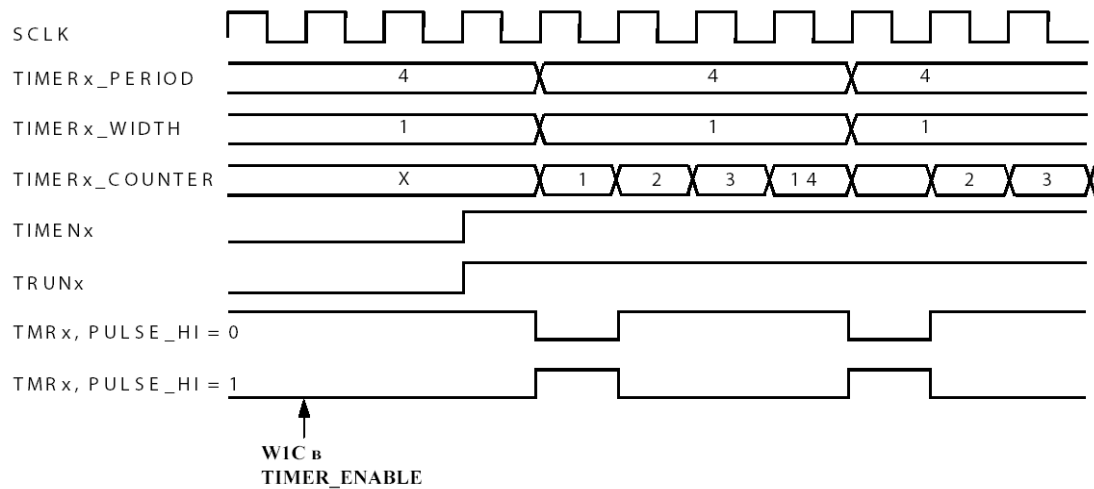


Рис. 15-9. Включение таймера

Пример временной диаграммы выключения таймера (режим PWM_OUT, PERIOD_CNT = 1)

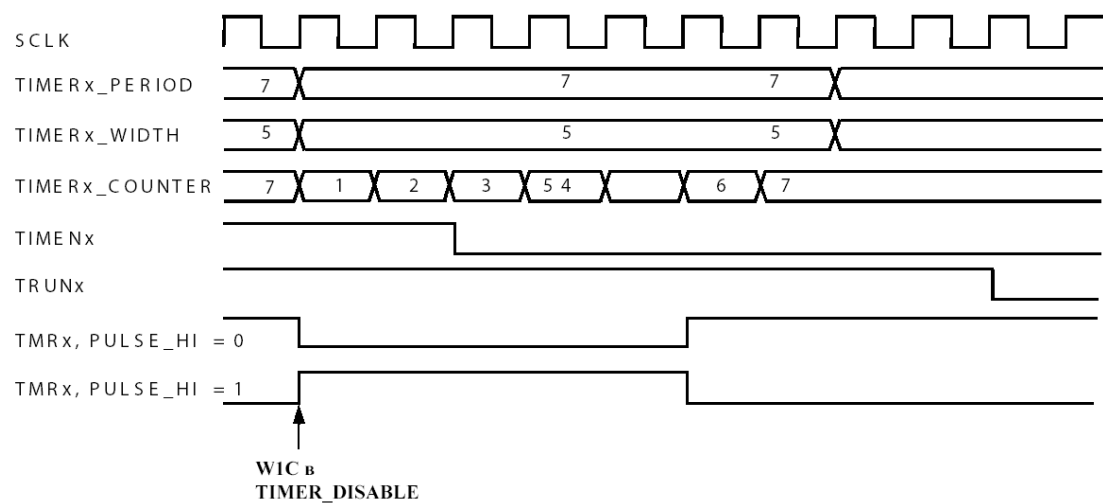


Рис. 15-10. Выключение таймера

Пример временной диаграммы включения и автоматического выключения таймера (режим PWM_OUT, PERIOD_CNT = 0)

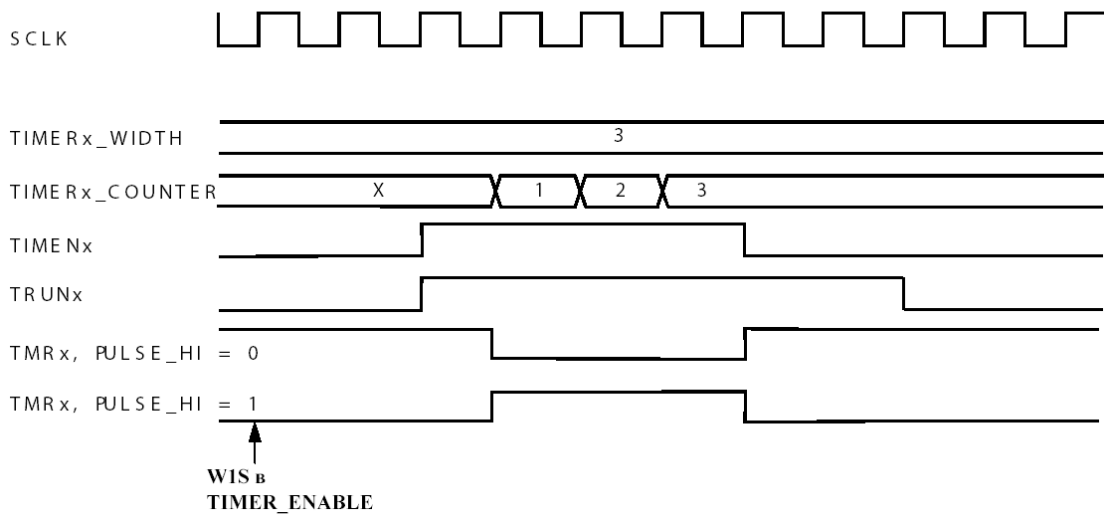


Рис. 15-11. Включение и автоматическое выключение таймера

Когда таймеры выключены, состояние регистров счётчика таймера сохраняется; при повторном включении таймера будет заново выполнена инициализация счётчика в соответствии с рабочим режимом. Регистры счётчика таймера доступны только для чтения. Программное изменение или предустановка счётчика таймера невозможны.

Режим широтно-импульсной модуляции (PWMOUT)

Режим PWMOUT включается при задании значения b#01 в поле TMODE регистра конфигурации таймера (TIMERx_CONFIG). В этом режиме вывод таймера TMRx работает как выход. Выходной сигнал на этом выводе может запрещаться установкой бита OUT_DIS в регистре конфигурации таймера.

В режиме PWM_OUT возможно независимое задание битов PULSE_HI, PERIOD_CNT, IRQ_ENA, OUT_DIS, CLK_SEL, EMU_RUN и TOGGLE_HI. Биты могут устанавливаться по отдельности или в любой комбинации; однако некоторые комбинации не имеют практического применения (например, TOGGLE_HI = 1 при OUT_DIS = 1 или PERIOD_CNT = 0).

После включения таймера в регистр счётчика загружается начальное значение. Если CLK_SEL = 0, счётчик начинает считать со значения 0x1. Если CLK_SEL = 1, счётчик сбрасывается в 0x0 аналогично режиму EXT_CLK. Таймер выполняет счёт вверх до значения, содержащегося в регистре периода таймера. При любом значении бита CLK_SEL, когда значение счётчика таймера совпадает со значением периода таймера, на следующем такте счётчик сбрасывается в 0x1.

В режиме PWM_OUT значение бита PERIOD_CNT определяет количество генерируемых импульсов. Когда бит PERIOD_CNT сброшен (режим одиночного импульса) таймер использует значение регистра TIMERx_WIDTH, генерирует

15 Таймеры

один импульс (один активный и один неактивный фронт), затем генерирует прерывание (если оно разрешено), после чего работа таймера останавливается. Когда `PERIOD_CNT` установлен (режим непрерывных импульсов), таймер использует значения регистров `TIMERx_PERIOD` и `TIMERx_WIDTH` и генерирует повторяющийся (и, возможно, модулированный) сигнал. Таймер генерирует прерывания (если они разрешены) в конце каждого периода и останавливается только после выключения. Если `PERIOD_CNT = 0`, выполняется счёт до значения длительности импульса; если `PERIOD_CNT = 1`, выполняется счёт до конца периода.

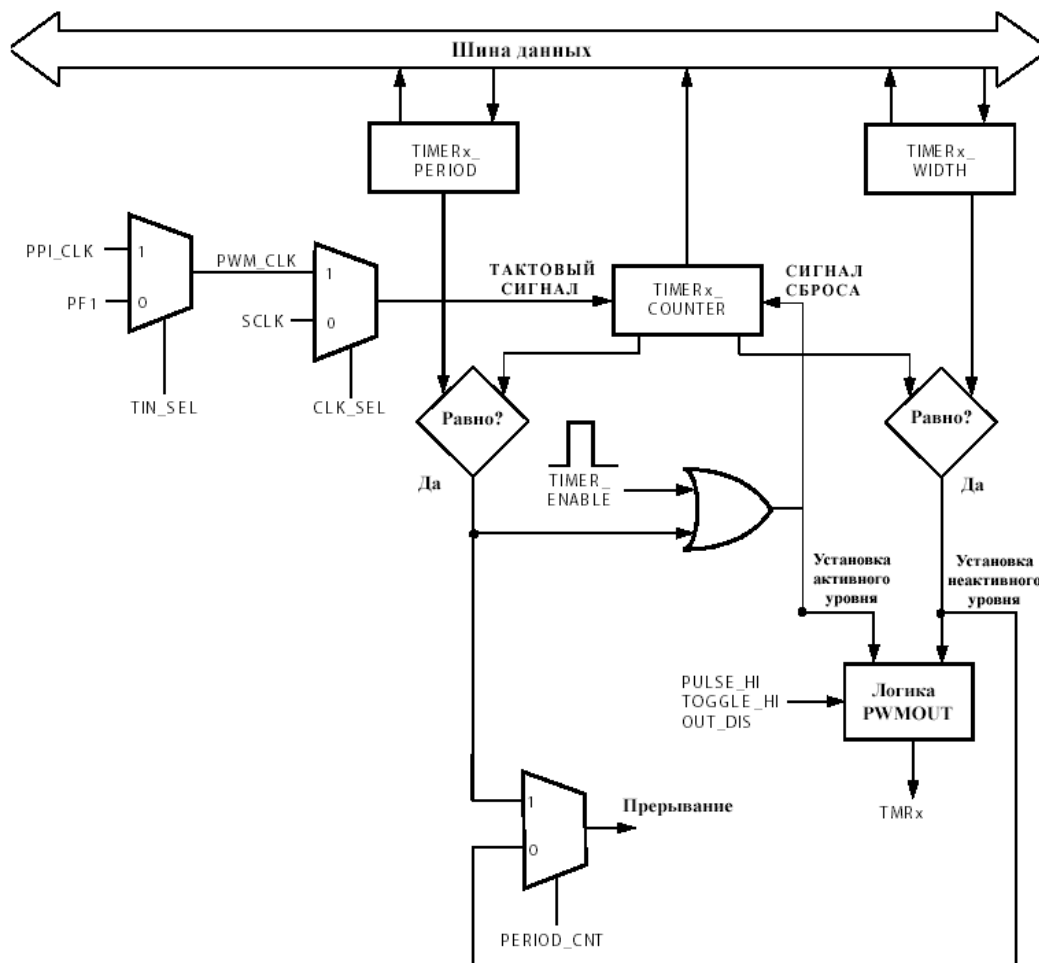


Рис. 15-12. Схема работы таймера, режим PWM_OUT

Отключение выходного вывода

Установка бита `OUT_DIS` регистра конфигурации таймера в режиме `PWM_OUT` запрещает выдачу сигнала на выходном выводе. При этом вывод `TMRx` переводится в третье состояние независимо от значений битов `PULSE_HI` и `TOGGLE_HI`. Это позволяет снизить потребление мощности, если выходной сигнал не используется.

Генерация одиночного импульса

Если в режиме PWM_OUT сброшен бит PERIOD_CNT, то таймер генерирует одиночный импульс на выводе TMRx. Это режим может использоваться, например, для реализации точной задержки. Длительность импульса определяется содержимым регистра длительности импульса; значение регистра периода в этом режиме не используется.

По окончании импульса устанавливается бит фиксации прерывания TIMIx и работа таймера автоматически останавливается. Если бит PULSE_HI установлен, то выводе TMRx выдаётся сигнал высокого уровня, если бит сброшен – сигнал низкого уровня.

Генерация сигнала с широтно-импульсной модуляцией

Если бит PERIOD_CNT установлен, то тактируемый внутренним сигналом таймер генерирует прямоугольные импульсы с заданным периодом и скважностью. В этом режиме также генерируются периодические прерывания для задач обработки сигналов в режиме реального времени.

В 32-разрядных регистрах периода (TIMERx_PERIOD) и длительности импульса (TIMERx_WIDTH) задаются значения периода счётчика и длительности импульса выходного сигнала с широтно-импульсной модуляцией (ШИМ-сигнала).

При работе таймера в режиме PWM_OUT сигнал на выводе TMRx переводится в неактивное состояние, когда значение счётчика равно значению регистра длительности импульса, и устанавливается в активное состояние, когда значение регистра периода уменьшается до нуля (или в начале работы таймера).

Для выбора активного уровня сигнала на выводе TMRx используется бит PULSE_HI в регистре TIMERx_CONFIG. Если бит сброшен, сигнал имеет активный низкий уровень. Если бит установлен, сигнал имеет активный высокий уровень. Когда таймер выключен в режиме PWM_OUT, на вывод TMRx выдаётся сигнал неактивного уровня.

В процессе работы таймера, прерывание таймера генерируется в конце каждого периода. В программе обслуживания прерывания должен сбрасываться бит фиксации прерывания (TIMIx) и может изменяться значение периода и/или длительности импульса. В приложениях, использующих ШИМ-сигналы, необходимо программно обновлять значения периода и длительности импульса во время работы таймера. Когда программа обновляет содержимое регистров периода или длительности импульса, до завершения текущего периода новые значения хранятся в специальных буферных регистрах. По окончании периода новые значения периода и длительности импульса становятся активны одновременно. Таким образом, выполняется запись новых значений регистров периода и длительности импульса, пока используются старые значения. Новые значения начинают использоваться, когда значение счётчика таймера

15 Таймеры

становится равным текущему значению периода. До завершения периода при чтении регистров периода и длительности импульса возвращаются старые значения.

Бит состояния `TOVF_ERRx` указывает на условие ошибки в режиме `PWM_OUT`. Бит `TOVF_ERRx` устанавливается, если при запуске значение периода равно нулю или единице или при обороте счётчика таймера. Он также устанавливается при обороте счётчика таймера, если значение регистра длительности импульса больше или равно значению регистра периода. Одновременно с установкой бита `TOVF_ERRx` обновляется поле `ERR_TYP`.

Для генерации сигнала максимальной частоты на выводе `TMRx` нужно установить значение периода, равное 2, а значение длительности импульса – 1. При этом сигнал на выводе `TMRx` будет переключаться на каждом такте сигнала `SCLK` (скважность 50%). Возможно задание любого значения периода в диапазоне от 2 до $(2^{32} - 1)$, включительно. Когда `PERIOD_CNT = 0`, возможно задание любого значения длительности импульса в диапазоне от 1 до $(2^{32} - 1)$, включительно.

Остановка работы таймера в режиме `PWM_OUT`

Во всех вариантах режима `PWM_OUT` таймер воспринимает операцию выключения (операцию `W1C` в регистр `TIMER_DISABLE`) как условие “ожидания остановки”. Когда таймер выключается, он автоматически завершает генерацию текущего импульса, после чего работа таймера прекращается. Это предотвращает усечение текущего импульса и нежелательные искажения PWM-сигнала на выводе `TMRx`. Процессор может определить момент остановки таймера при помощи опроса соответствующего бита `TRUNx` в регистре `TIMER_STATUS` (если таймер остановлен, при чтении бита возвращается 0) или, ожидая последнее прерывание (если генерация прерываний разрешена). Необходимо отметить, что пока таймер не будет остановлен и при чтении `TRUNx` не будет возвращён 0, задание новой конфигурации (запись нового значения в регистр `TIMERx_CONFIG`) таймера невозможно.

В режиме генерации одиночного импульса `PWM_OUT` (`PERIOD_CNT = 0`) для остановки работы таймера необязательно выполнять запись в регистр `TIMER_DISABLE`. В конце генерации импульса таймер останавливается автоматически, сбрасывается соответствующий бит в регистре `TIMER_ENABLE` (и `TIMER_DISABLE`) и бит `TRUNx` (см. рис. 15-11). Для генерации нескольких импульсов, необходимо записать единицу в `TIMER_ENABLE`, дождаться остановки таймера, снова записать единицу в `TIMER_ENABLE` и т.д.

При необходимости, в режиме `PWM_OUT` возможна принудительная мгновенная остановка работы таймера. Для этого необходимо сначала записать единицу в соответствующий бит регистра `TIMER_DISABLE`, а затем записать единицу в бит `TRUNx` регистра `TIMER_STATUS`. При этом работа таймера останавливается независимо от того, ожидает ли он завершения текущего

15 Таймеры

периода ($PERIOD_CNT = 1$) или текущего импульса ($PERIOD_CNT = 0$). Это свойство может использоваться для мгновенного восстановления контроля над таймером при выполнении процедуры исправления ошибки.

- ⊘ При использовании этого свойства следует соблюдать осторожность, так как при этом может исказиться форма ШИМ-сигнала на выводе $TMRx$.

В режиме PWM_OUT с непрерывной генерацией импульсов ($PERIOD_CNT = 1$) каждый таймер опрашивает бит $TIMENx$ в конце каждого периода. Работа таймера останавливается в конце первого периода, в котором $TIMENx$ равен нулю. При этом (если не выполняется операция $W1C$ в биты $TRUNx$), если выключить и снова включить таймер до завершения текущего периода, то работа таймера продолжится в обычном режиме. Обычно, в программе выполняется выключение таймера и затем ожидается остановка его работы. Когда $PERIOD_CNT = 0$ работа таймера всегда останавливается по окончании первого импульса.

Внешнее тактирование в режиме PWM_OUT

По умолчанию таймер работает от внутреннего тактового сигнала $SCLK$. Однако, если установлен бит CLK_SEL в регистре конфигурации таймера ($TIMERx_CONFIG$), таймер тактируется внешним сигналом PWM_CLK . Обычно сигнал PWM_CLK поступает с вывода $PF1$; он также может поступать с вывода PPI_CLK , когда таймеры настроены для работы с PPI -портом. В зависимости от конфигурации на входы PWM_CLK различных таймеров могут подаваться различные сигналы. Бит $PERIOD_CNT$ в режиме PWM_OUT задаёт генерацию либо ШИМ-сигналов, либо одиночного импульса, длительность которого задаётся регистром $TIMERx_WIDTH$.

Когда установлен бит CLK_SEL , счётчик сбрасывается в $0x0$ при запуске и инкрементируется по каждому переднему фронту PWM_CLK . Изменения сигнала на выводе $TMRx$ происходят по переднему фронту PWM_CLK . Выбор заднего фронта PWM_CLK для тактирования таймера невозможен. В этом режиме бит $PULSE_HI$ определяет только полярность формируемых импульсов. Прерывание таймера может незначительно опережать появление соответствующего фронта сигнала на выводе $TMRx$ (так как прерывание генерируется по фронту сигнала $SCLK$, а изменение сигнала на выводе $TMRx$ происходит по фронту PWM_CLK , возникающему позже). При этом новое значение периода и длительности импульса может быть задано сразу после генерации прерывания. По окончании периода происходит оборот счётчика и его значение становится равным $0x1$.

Скважность тактового сигнала PWM_CLK не обязательно равна 50%; однако минимальные времена, в течение которых тактовый сигнал имеет низкий или высокий уровень, равны одному периоду $SCLK$. При этом максимальная частота тактового сигнала PWM_CLK составляет $SCLK/2$.

Вывод PF1 может являться источником тактового сигнала таймера только, если он настроен на вход. Когда любой из таймеров работает в режиме PWM_OUT при CLK_SEL = 1 и TIN_SEL = 0, бит PF1 в регистре FIO_DIR игнорируется и вывод PF1 принудительно настраивается на вход.

Режим инверсии PULSE_HI

Время, в течение которого сигнал, формируемый в режиме PWM_OUT при PERIOD_CNT = 1, имеет активный уровень, обычно фиксировано; время, в течение которого сигнал имеет неактивный уровень, программно задаётся (при помощи регистра TIMERx_WIDTH). Когда два таймера работают синхронно и имеют одинаковые значения периодов, импульсы выровнены по активному фронту, как показано на рис. 15-13.

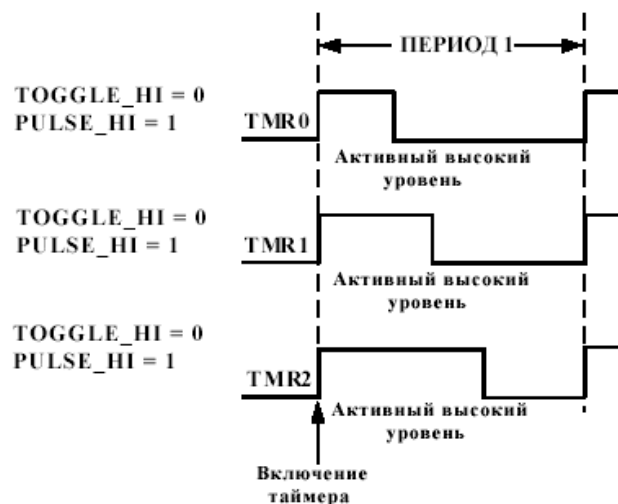


Рис. 15-13. Диаграмма генерации таймерами импульсов, выровненных по активному фронту.

Режим TOGGLE_HI позволяет управлять временными параметрами активного и неактивного фронтов формируемого выходного сигнала. Фаза между активными фронтами выходных сигналов двух таймеров может задаваться программно. Состояние бита PULSE_HI в этом режиме изменяется на каждом периоде. Смежные импульсы активного высокого и активного низкого уровня, представляют собой две половины полностью произвольного прямоугольного сигнала. Формируемый сигнал имеет активный высокий уровень, когда бит PULSE_HI установлен, и активный низкий уровень, когда PULSE_HI сброшен. Значение бита TOGGLE_HI не имеет значения, пока таймер не работает в режиме PWM_OUT и PERIOD_CNT не равен единице.

В режиме TOGGLE_HI, когда PULSE_HI установлен, импульс активного низкого уровня генерируется в нечётных периодах, а импульс активного высокого уровня – в чётных периодах. Когда PULSE_HI сброшен, импульс

15 Таймеры

активного высокого уровня генерируется в нечётных периодах, а импульс активного низкого уровня – в чётных периодах.

Неактивное состояние сигнала в конце одного периода совпадает с активным состоянием в начале следующего периода, таким образом, изменения уровня выходного сигнала происходят только, когда значение счетчика равно длительности импульса. В результате выходной сигнал имеет форму импульсов, повторяющихся каждые два периода счетчика и центрированных относительно конца первого периода (или начала второго периода).

На рис. 15-14 показан пример, в котором все три таймера имеют одинаковые значения периодов. Если в программе параметры ШИМ-сигнала не изменяются в процессе работы таймера, скважность сигнала равна 50%. Значения регистров `TIMERx_WIDTH` определяют фазу между сигналами.

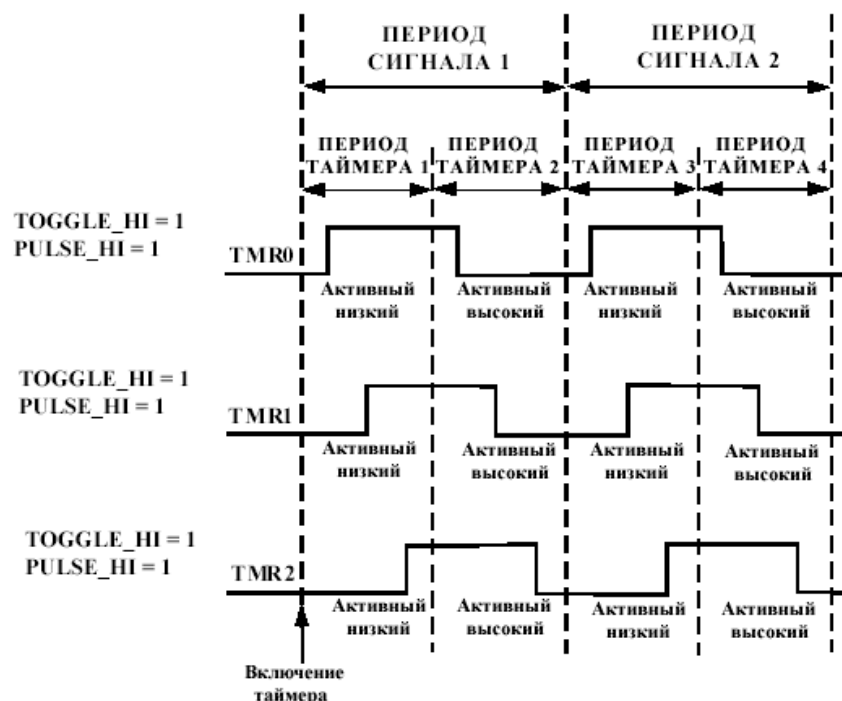


Рис. 15-14. Диаграмма работы трёх таймеров с одинаковыми значениями периода

Аналогичным образом, при помощи двух таймеров могут генерироваться неперекрывающиеся тактовые сигналы. Для этого импульсы выравняются по середине периода, и инвертируется полярность сигнала одного из таймеров (см. рис. 15-15).

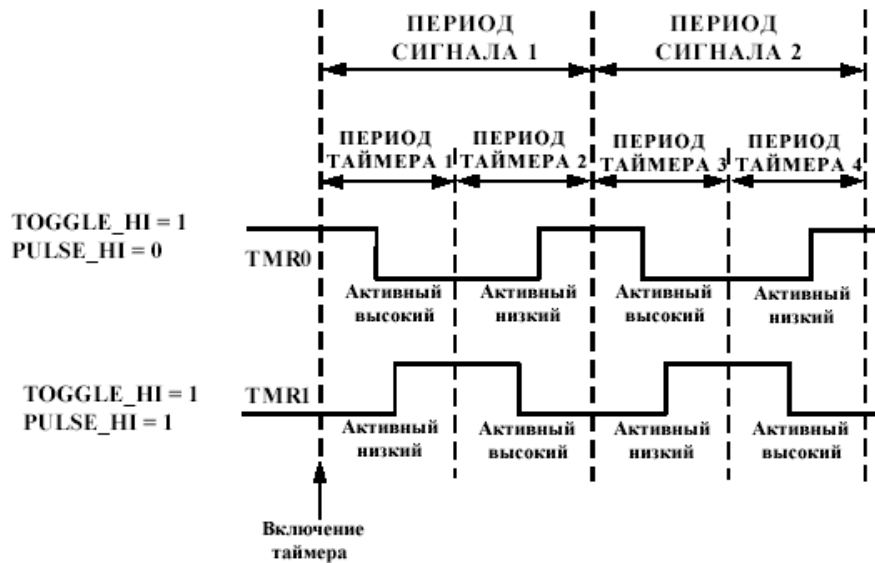


Рис. 15-15. Диаграмма работы двух таймеров, формирующих неперекрывающиеся тактовые сигналы

При `TOGGLE_HI = 0` программа обновляет значения регистров периода и длительности импульса один раз за период формируемого сигнала. При `TOGGLE_HI = 1` программа обновляет значения регистров периода и длительности импульса дважды за период формируемого сигнала; при этом в них записываются вдвое меньшие значения. В четных периодах в регистр длительности импульса вместо значения длительности импульса записывается разность периода и длительности импульса. Это делается для того, чтобы получить выровненные по середине периода импульсы.

Далее проводится псевдокод для формирования сигналов при `TOGGLE_HI = 0`:

```
int period, width;
for (;;) {
    period = generate_period(...);
    width = generate_width(...);

    wait for (interrupt);
    write(TIMERx_PERIOD, period);
    write(TIMERx_WIDTH, width);
}
```

При `TOGGLE_HI = 1` псевдокод принимает следующий вид:

```
int period, width;
int per1, per2, wid1, wid2;

for (;;) {
    period = generate_period(...);
    width = generate_width(...);
```

```
per1 = period/2;
wid1 = width/2;

per2 = period/2;
wid2 = width/2;

waitfor (interrupt);

write(TIMERx_PERIOD, per1);
write(TIMERx_WIDTH, per1 - wid1);

waitfor (interrupt);

write(TIMERx_PERIOD, per2);
write(TIMERx_WIDTH, wid2);

}
```

Как показано в этом примере, формируемые импульсы не обязательно симметричны ($wid1$ не обязательно равно $wid2$). Для задания фазы между формируемыми импульсами можно изменять значение периодов ($per1$ не обязательно равен $per2$).

В режиме `TOGGLE_HI` бит `TRUNx` регистра `TIMER_STATUS` обновляется только в конце четных периодов. При записи единицы в бит регистра `TIMER_DISABLE`, текущая пара периодов счётчика (один период сигнала) завершается до выключения таймера.

Как и при `TOGGLE_HI = 0` биты ошибки устанавливаются если:
`TIMERx_WIDTH >= TIMERx_PERIOD`, `TIMERx_PERIOD = 0` или
`TIMERx_PERIOD = 1`.

Режим измерения периода и длительности импульсов (WDTH_CAP)

В режиме `WDTH_CAP` вывод `TMRx` работает как вход. Тактируемый внутренним сигналом таймер используется для определения периода и длительности импульса внешнего прямоугольного сигнала. Данный режим активизируется записью `b#10` в поле `TMODE` в регистре конфигурации таймера (`TIMERx_CONFIG`).

При включении таймера в этом режиме значение счётчика таймера (регистра `TIMERx_COUNTER`) сбрасывается в `0x0000 0001` и счет не начинается до тех пор, пока не будет обнаружен передний фронт сигнала на выводе `TMRx`.

Когда таймер обнаруживает первый передний фронт, начинается инкремент счётчика. При обнаружении заднего фронта сигнала таймер помещает текущее

15 Таймеры

32-разрядное значение регистра `TIMERx_COUNTER` в буферный регистр длительности импульса. По следующему переднему фронту таймер передает текущее 32-разрядное значение регистра `TIMERx_COUNTER` в буферный регистр периода. При этом значение регистра счетчика снова сбрасывается в `0x0000 0001`, продолжается счет счетчика и захват значений длительности и периода импульсов до тех пор, пока таймер не будет выключен.

При работе в этом режиме программа может измерять и длительность импульса, и период сигнала. Для задания того, какой из фронтов сигнала на выводе `TMRx` является передним, а какой задним, используется бит `PULSE_HI` в регистре `TIMERx_CONFIG`. Если бит `PULSE_HI` сброшен, измерение инициируется по заднему фронту импульса, значение регистра счетчика записывается в буфер длительности импульса по переднему фронту импульса и значение периода захватывается по следующему заднему фронту импульса. Когда бит `PULSE_HI` установлен, измерение инициируется передним фронтом импульса, значение регистра счетчика записывается в буферный регистр длительности импульса по заднему фронту импульса и значение периода захватывается по следующему переднему фронту импульса.

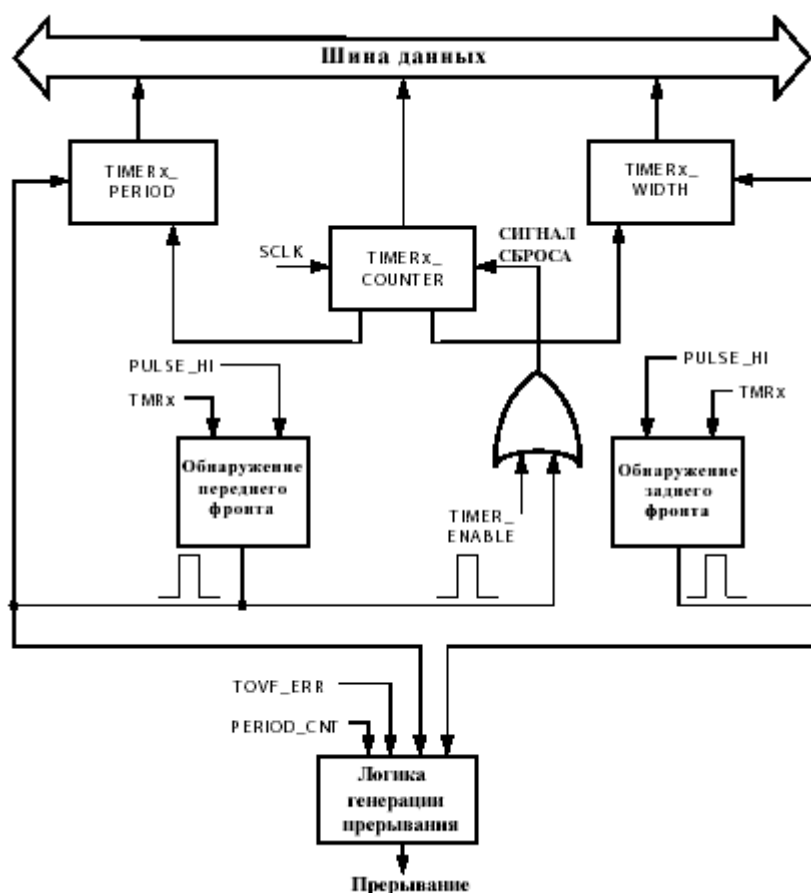


Рис. 15-16. Схема работы таймера в режиме `WDTM_CAP`

Следующие три события в режиме `WDTM_CAP` происходят одновременно:

15 Таймеры

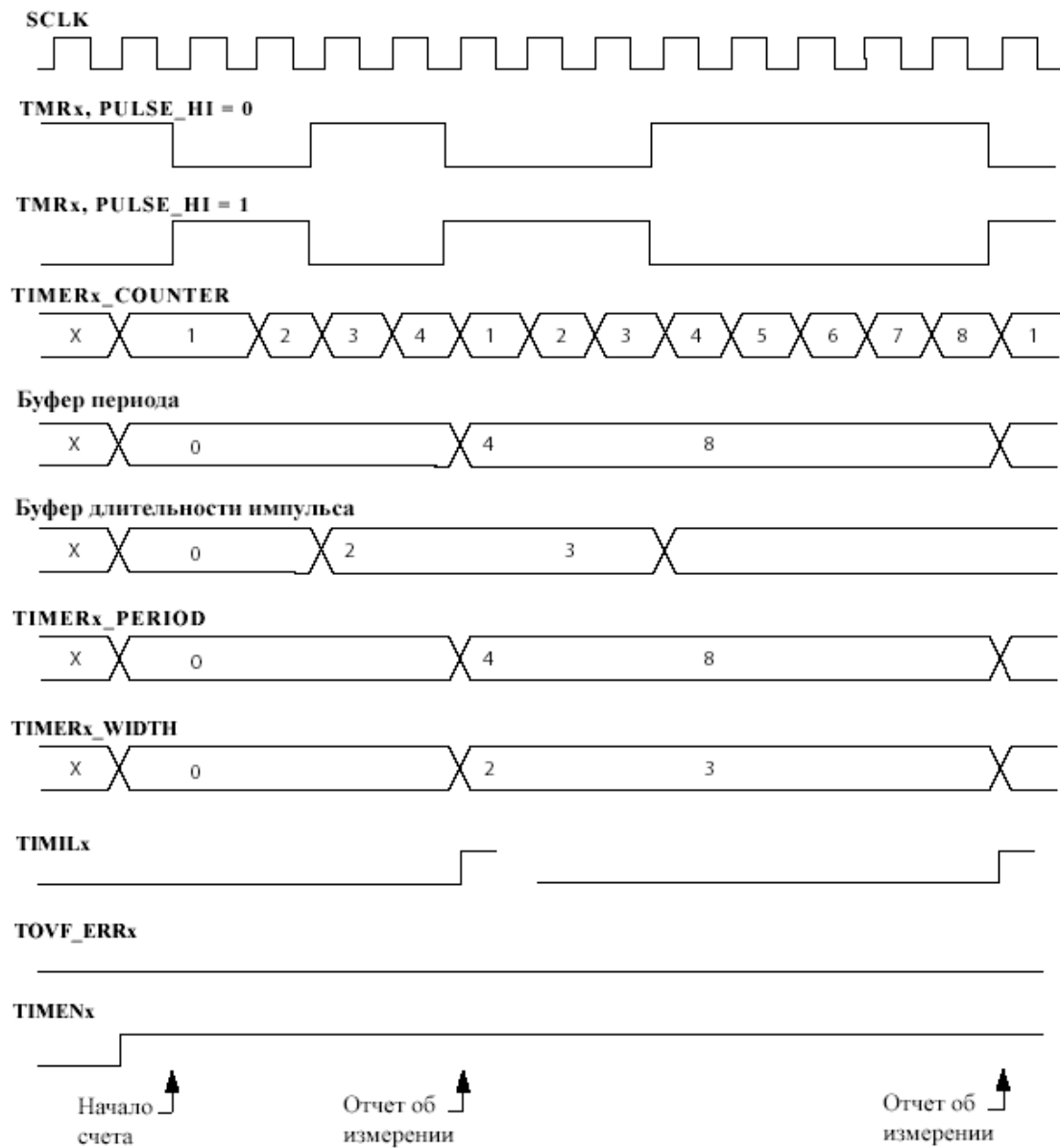
1. Содержимое регистра `TIMERx_PERIOD` обновляется значением буферного регистра периода.
2. Содержимое регистра `TIMERx_WIDTH` обновляется значением буферного регистра длительности импульса.
3. Устанавливается бит фиксации прерывания таймера `TIMILx` (если генерация прерывания разрешена), но не генерируется ошибка.

Момент времени, в который выполняется этот набор действий, задается битом `PERIOD_CNT` в регистре `TIMERx_CONFIG`. Вместе эти три события называются отчетом об измерении. В момент отчета об измерении бит фиксации ошибки переполнения счетчика таймера (`TOVF_ERRx`) не устанавливается. Отчет об измерении происходит не более одного раза за период входного сигнала.

Текущее значение счетчика всегда копируется в буфер длительности импульса и буфер периода по переднему и заднему фронтам входного сигнала, соответственно. Однако эти значения недоступны программе. При событии отчета об измерении захваченные значения передаются в доступные программе регистры, и устанавливается прерывание таймера, сигнализирующее, что значения в регистрах `TIMERx_PERIOD` и `TIMERx_WIDTH` доступны для чтения. Когда бит `PERIOD_CNT` установлен, отчет об измерении выполняется сразу после помещения нового значения в буферный регистр периода (по переднему фронту). Когда бит `PERIOD_CNT` сброшен, отчет об измерении выполняется сразу после помещения нового значения в буферный регистр длительности импульса (по заднему фронту).

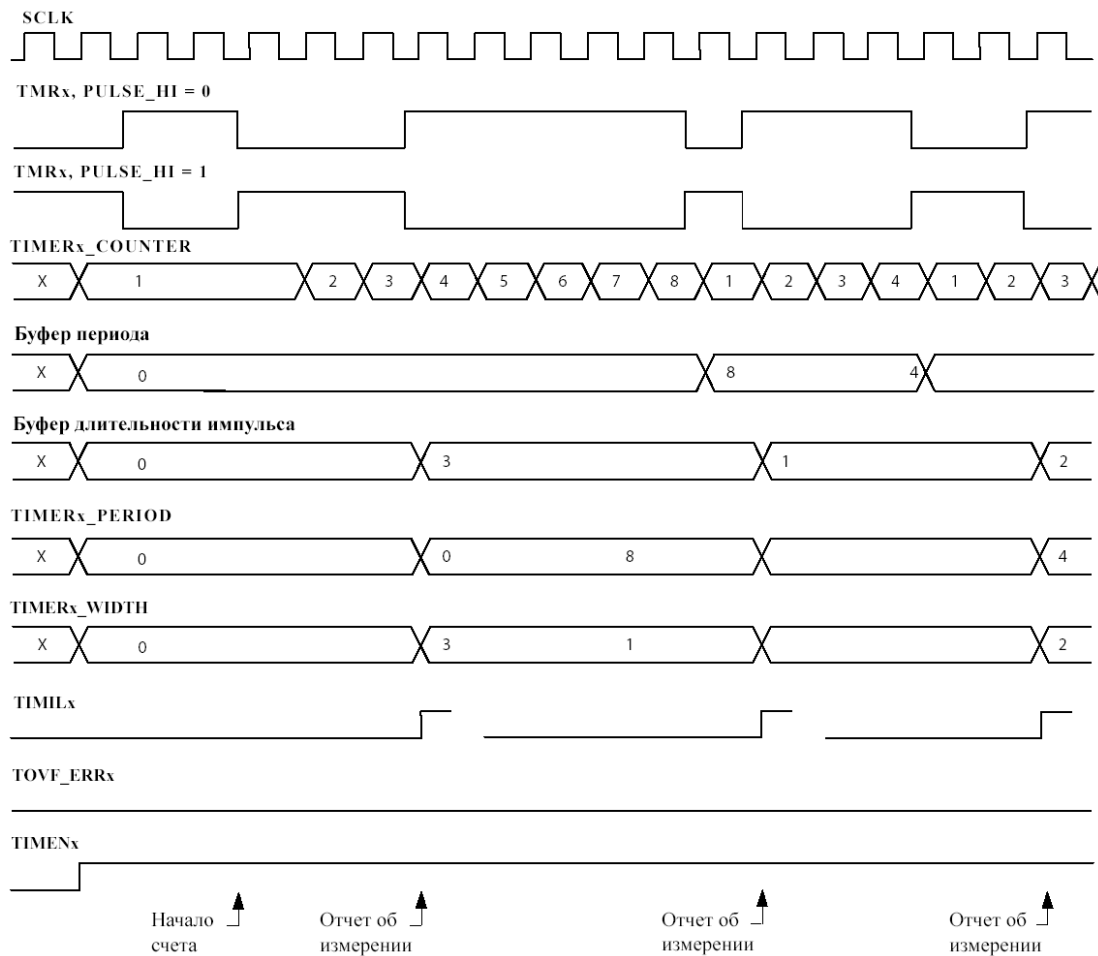
Если бит `PERIOD_CNT` установлен и поступает передний фронт (см. рис. 15-17), то в регистрах `TIMERx_PERIOD` и `TIMERx_WIDTH` будут содержаться значения периода и длительности импульса, измеренные в предыдущем периоде. Если бит `PERIOD_CNT` сброшен и поступает задний фронт (см. рис. 15-18), то в регистре `TIMERx_WIDTH` содержится значение длительности импульса, измеренное в текущем периоде, а в регистре `TIMERx_PERIOD` содержится значение периода импульса, измеренное в предыдущем периоде.

15 Таймеры



Примечание: Для простоты задержка синхронизации между фронтами сигнала TMRx и обновлением буферных регистров не показана.

Рис. 15-17. Пример временной диаграммы отчета об измерении периода (режим WDTM_CAP, PERIOD_CNT = 1)



Примечание: Для простоты задержка синхронизации между фронтами сигнала TMRx и обновлением буферных регистров не показана.

Рис. 15-18. Пример временной диаграммы отчета об измерении длительности импульса (режим WIDTH_CAP, PERIOD_CNT = 0)

Если бит PERIOD_CNT сброшен и поступает первый передний фронт, то первое значение периода еще не было измерено и значение, формируемое в отчете об измерении, недостоверно. В этом случае при чтении регистра TIMERx_PERIOD возвращается 0, как показано на рис. 15-18. Для измерения длительности импульса сигнала, имеющего только один передний и только один задний фронт, следует установить PERIOD_CNT = 0. Если в данном случае PERIOD_CNT = 1, в буферный регистр периода значение периода помещено не будет. Вместо этого при переполнении счетчика будет сгенерировано прерывание ошибки (если оно разрешено). В данном случае при чтении регистров TIMERx_WIDTH и TIMERx_PERIOD возвращается 0 (так как отчет об измерении не происходит и значение буферного регистра длительности импульса не копируется в TIMERx_WIDTH). См. первое прерывание на рис. 15-19.



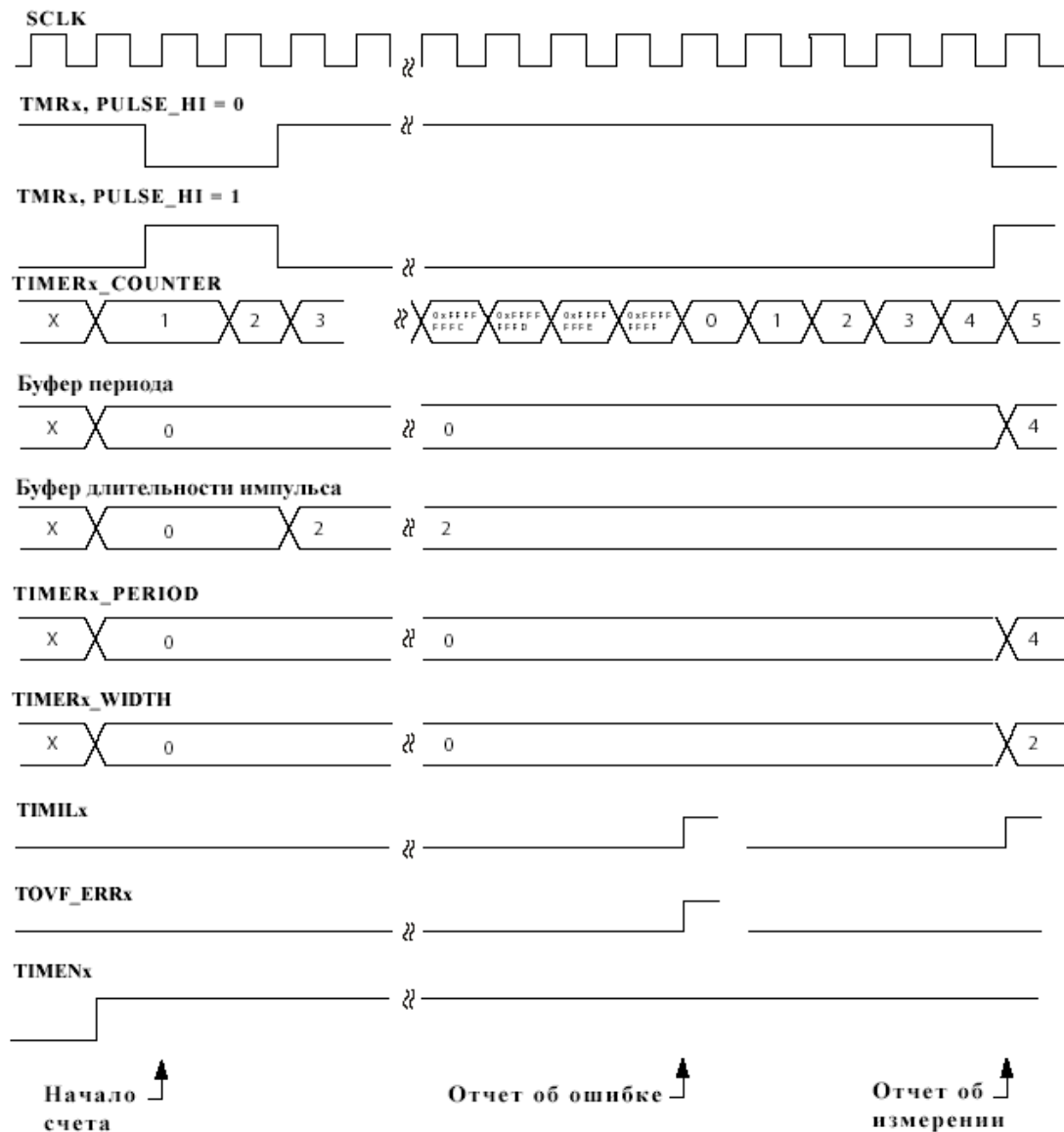
При использовании режима PERIOD_CNT = 0, описанного выше, для измерения длительности одиночного импульса рекомендуется выключать таймер после поступления прерывания в конце интервала

15 Таймеры

измерения. При желании затем можно снова включить таймер для проведения следующего измерения. Эта процедура предотвращает вход таймера в режим непрерывного счёта после измерения длительности импульса и возникновения ошибок, вызванных переполнением счетчика таймера.

Прерывание таймера (если оно разрешено) генерируется при переходе значения регистра счетчика таймера из 0xFFFF FFFF в 0 в отсутствие переднего фронта сигнала. В этот момент устанавливается бит `TOVF_ERRx` в регистре `TIMER_STATUS` и бит `ERR_TYP` в регистре `TIMERx_CONFIG`, сигнализирующие о переполнении счетчика вследствие превышения периодом сигнала диапазона значений счетчика. Эта совокупность событий называется отчетом об ошибке. Таймер генерирует прерывание в режиме `WDTH_CAP` либо при возникновении ошибки (отчет об ошибке), либо при готовности новых измеренных значений для чтения (отчет об измерении). Одновременное возникновение отчета об ошибке и отчета об измерении невозможно. Регистры `TIMERx_PERIOD` и `TIMERx_WIDTH` никогда не обновляются при возникновении отчета об ошибке. Дополнительную информацию см. на рис. 15-19 и 15-20.

15 Таймеры



Примечание: Для простоты задержка синхронизации между фронтами сигнала TMRx и обновлением буферных регистров не показана.

Рис. 15-19. Пример временной диаграмм при переполнении счётчика, сопровождаемом измерением периода (режим WDM_CAP, PERIOD_CNT = 1)

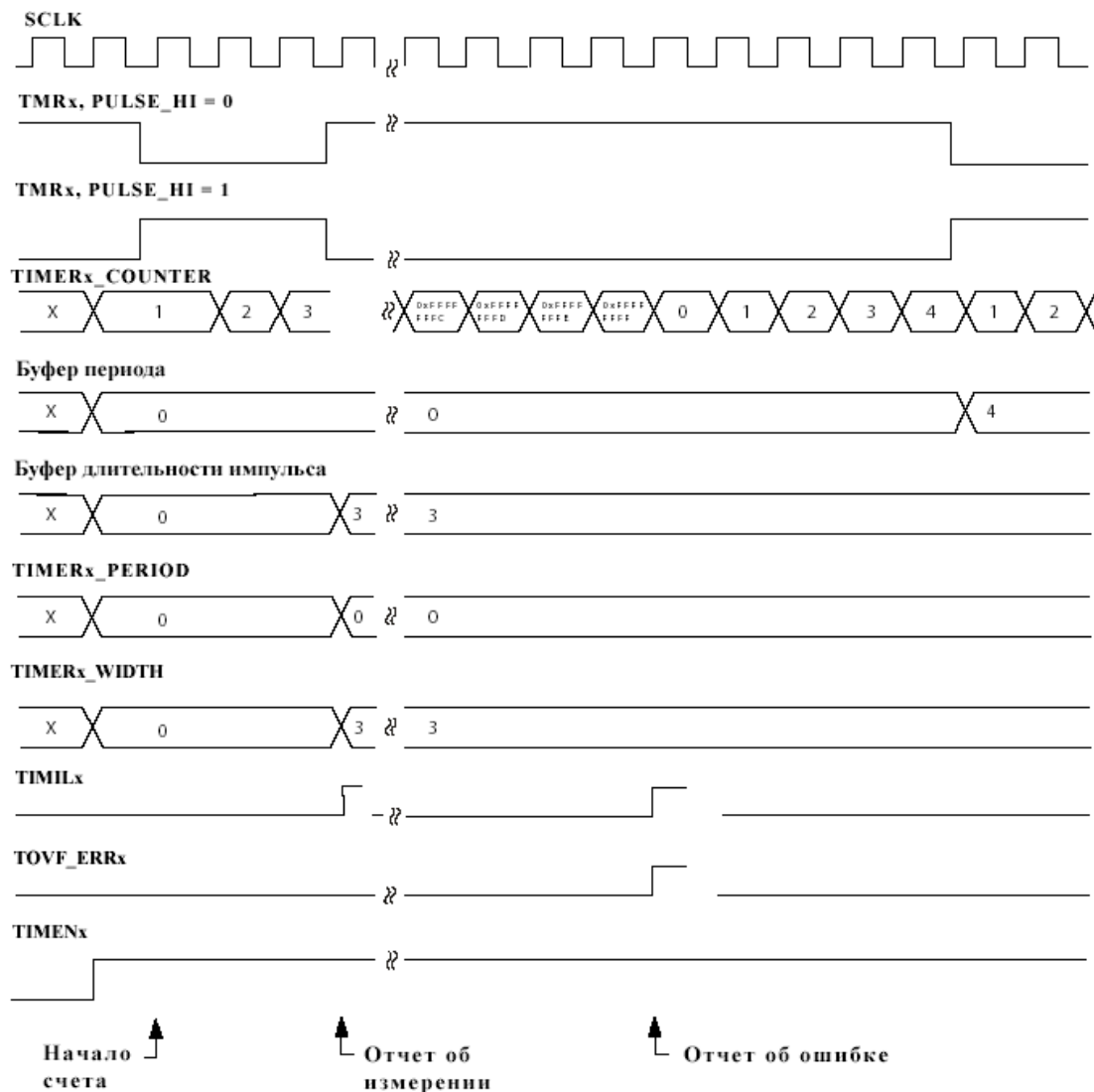


Рис. 15-20. Пример временных диаграмм при измерении периода, сопровождаемом переполнением счётчика (режим WDTH_CAP, PERIOD_CNT = 0)

Биты TIMIx и TOVF_ERRx являются залипающими; они должны сбрасываться программно. Если возникает переполнение таймера и PERIOD_CNT = 1, содержимое регистров TIMERx_PERIOD и TIMERx_WIDTH не обновляется. Если возникает переполнение таймера и PERIOD_CNT = 0, содержимое регистров TIMERx_PERIOD и TIMERx_WIDTH обновляется только, если при предыдущем отчете об измерении был обнаружен задний фронт.

Для измерения периодов входных сигналов, превышающих значение 0xFFFF FFFF, в программе можно выполнять подсчет числа прерываний отчета об ошибке, возникающих между прерываниями отчета об измерении. Каждое прерывание отчета об ошибке добавляет к значению периода 2^{32} тактов SCLK, однако значение длительности импульса при этом восстановить невозможно. Например, на рис. 15-19 период равен 0x10000 0004, а длительность импульса может быть равна 0x00000 0002 или 0x10000 0002.

15 Таймеры

Скважность сигнала, подаваемого на вывод TMRx, не обязательно равна 50%; однако минимальные времена, в течение которых этот сигнал имеет низкий или высокий уровень, равны одному периоду SCLK. При этом максимальная частота сигнала, поступающего на вход TMRx, составляет SCLK/2. При этих условиях в режиме WDN_CAP таймер измерит следующие значения: период = 2, Длительность импульса = 1.

Режим автоматического измерения скорости передачи

Любой из трех таймеров может обеспечивать функцию автоматического определения скорости двоичной передачи UART-порта. При установке бита выбора входа таймера (TIN_SEL) в режиме WDN_CAP вместо опроса вывода TMRx таймер опрашивает вывод приема данных UART-порта (RX).

⊘ Не следует разрешать работу UART-порта до завершения автоматического определения скорости передачи.

Длительность импульсов в последовательном потоке данных может быть определена программно. Вследствие того, что работа таймера синхронна с операцией UART-порта (тактовые сигналы обоих устройств формируются из сигнала схемы PLL), длительность импульсов может использоваться для расчета делителя скорости двоичной передачи UART-порта по формуле.

Делитель = ((TIMERx_WIDTH) / (16 × Число измеренных битов UART))

Не рекомендуется выполнять измерение длительности импульса по одиночному биту. Вместо этого следует проводить измерения по большему количеству битов, что позволит увеличить количество циклов счётчика и, следовательно, повысить разрешение. Обычно для автоматического определения скорости передачи используется символ NULL (ASCII 0x00), показанный на рис. 15-21.



Рис. 15-21. Автоматическое измерение скорости передачи по символу 0x00

Так как кадр, приведённый на рис. 15-21, включает 8 битов данных и 1 старт-бит, формула принимает следующий вид:

15 Таймеры

$$\text{Делитель} = (\text{TIMERx_WIDTH}) / (16 \times 9)$$

Реальные сигналы UART-порта обычно имеют асимметричные передние и задние фронты; кроме того, логический уровень опроса сигналов не всегда в точности совпадает с серединой диапазона напряжения. При больших скоростях передачи, автоматическое определение скорости передачи, основанное на измерении длительности импульса, без использования дополнительных ограничений аналоговых сигналов может давать неадекватные результаты. Настоятельно рекомендуется измерять период сигнала, что позволяет повысить точность расчета.

Например, для автоматического определения скорости передачи может использоваться ASCII символ “@” (40h) и измеряться период между двумя последовательными задними фронтами. Как показано на рис. 15-22, при этом измеряется период между задним фронтом старт-бита и задним фронтом, возникающем после передачи бита 6. Так как данный период включает 8 бит, используйте следующую формулу:

$$\text{Делитель} = (\text{TIMERx_PERIOD}) / (16 \times 8)$$

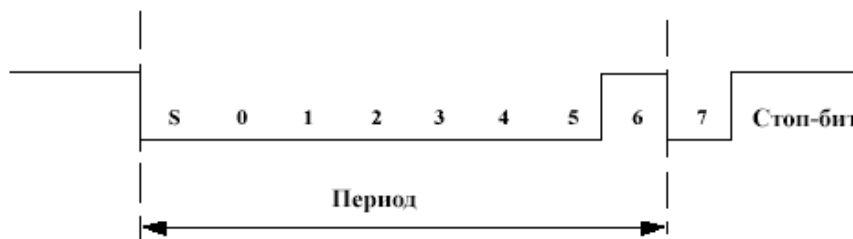


Рис. 15-22. Автоматическое измерение скорости передачи по символу 0x40

Режим счётчика внешних воздействий (EXT_CLK)

В режиме EXT_CLK вывод TMRx является входом. В этом режиме таймер работает как счетчик, тактируемый сигналом от внешнего источника, который может быть асинхронным по отношению к тактовому сигналу системы. Текущее содержимое регистра TIMERx_COUNTER отражает количество обнаруженных передних фронтов сигнала. Этот режим включается при записи значения b#11 в поле TMODE в регистре TIMERx_CONFIG. В регистр TIMERx_PERIOD записывается максимальное значение периода внешнего сигнала.

Скважность сигнала, подаваемого на вывод TMRx, не обязательно равна 50%; однако минимальные времена, в течение которых этот сигнал имеет низкий или высокий уровень, равны одному периоду SCLK. При этом максимальная частота сигнала, поступающего на вход TMRx, составляет SCLK/2.

15 Таймеры

Возможно задание любого значения периода от 1 до $(2^{32} - 1)$ включительно.

После включения таймера содержимое регистра счётчика таймера сбрасывается в 0x0 и таймер ждет появления первого активного фронта сигнала на выводе TMRx. По этому фронту регистр счетчика инкрементируется (после чего содержит значение 0x01). Затем счетчик таймера инкрементируется по каждому активному фронту. После того, как значение счетчика станет равным значению периода, устанавливается бит TIM1Lx и генерируется прерывание. По следующему активному фронту сигнала на выводе TMRx в регистр счетчика снова загружается значение 0x1. Счётчик продолжает считать до того, как будет выключен таймер. Бит PULSE_HI определяет, какой из фронтов является активным. Если PULSE_HI установлен, активным является передний фронт, если PULSE_HI сброшен – задний.

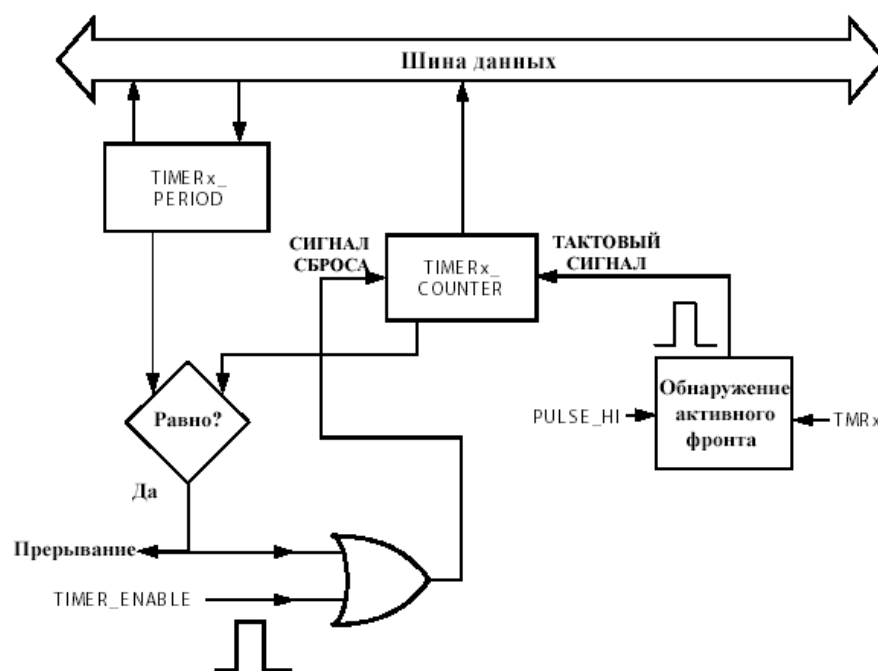


Рис. 15-23. Схема работы таймера в режиме EXT_CLK

В этом режиме значения битов конфигурации TIN_SEL и PERIOD_CNT ни на что не влияют. Биты TOVF_ERRx и ERR_TYP устанавливаются в следующих случаях: когда значение регистра счётчика изменяется из 0xFFFF FFFF в 0, когда при запуске таймера значение регистра периода равно нулю или когда значение регистра счётчика изменяется из значения периода в 0. В этом режиме регистр длительности импульса не используется.

Использование таймеров совместно с PPI-портом

В определённых режимах работы PPI-порта возможно использование до двух таймеров для генерации сигналов кадровой синхронизации. Дополнительные

указания по настройке таймеров для использования совместно с PPI-портом см. в разделе “Кадровая синхронизация в режимах общего назначения” главы 11.

Прерывания

Каждый из трёх таймеров может генерировать одно прерывание. Три генерируемых сигнала прерывания направляются в блок контроллера прерываний системы, осуществляющий назначение приоритетов и маскирование. Программа может определить источник прерывания при помощи регистра состояния таймеров (TIMER_STATUS), в котором фиксируются прерывания таймеров. Биты фиксации прерываний имеют тип WIC и должны быть сброшены в программе обслуживания прерывания до вызова команды RTI, чтобы исключить повторное срабатывание прерывания.

Для разрешения генерации прерывания необходимо установить бит IRQ_ENA и снять маскирование источника прерывания в регистре маскирования прерываний системы (SIC_IMASK). Для опроса бита TIMILx без генерации прерывания, следует установить IRQ_ENA и оставить прерывание маскированным. Запросы прерываний также генерируются (если установлен бит IRQ_ENA) условиями ошибок.

Контроллер прерываний системы позволяет организовать гибкую обработку прерываний. Таймеры могут совместно использовать один канал прерывания; при этом одна программа обслуживания прерывания обслуживает несколько таймеров. В режиме PWM несколько таймеров могут работать с одинаковым значением периода и одновременно формировать запросы прерываний. В данном случае в программе обслуживания прерывания должен выполняться одновременный сброс битов фиксации прерывания путём записи значения 0x07 в регистр TIMER_STATUS.

Если генерация прерываний разрешена, программа обслуживания прерывания должна сбрасывать бит TIMILx регистра TIMERx_STATUS до выполнения команды RTI, чтобы исключить повторное срабатывание прерывания. При этом необходимо учитывать задержки записи в регистры системы. Если между командой сброса TIMILx и командой RTI выполняется небольшое число команд, может потребоваться добавление дополнительной команды SSYNC. В режиме EXT_CLK бит TIMILx должен сбрасываться в самом начале программы обслуживания прерывания; это предотвращает пропуск внешних воздействий.

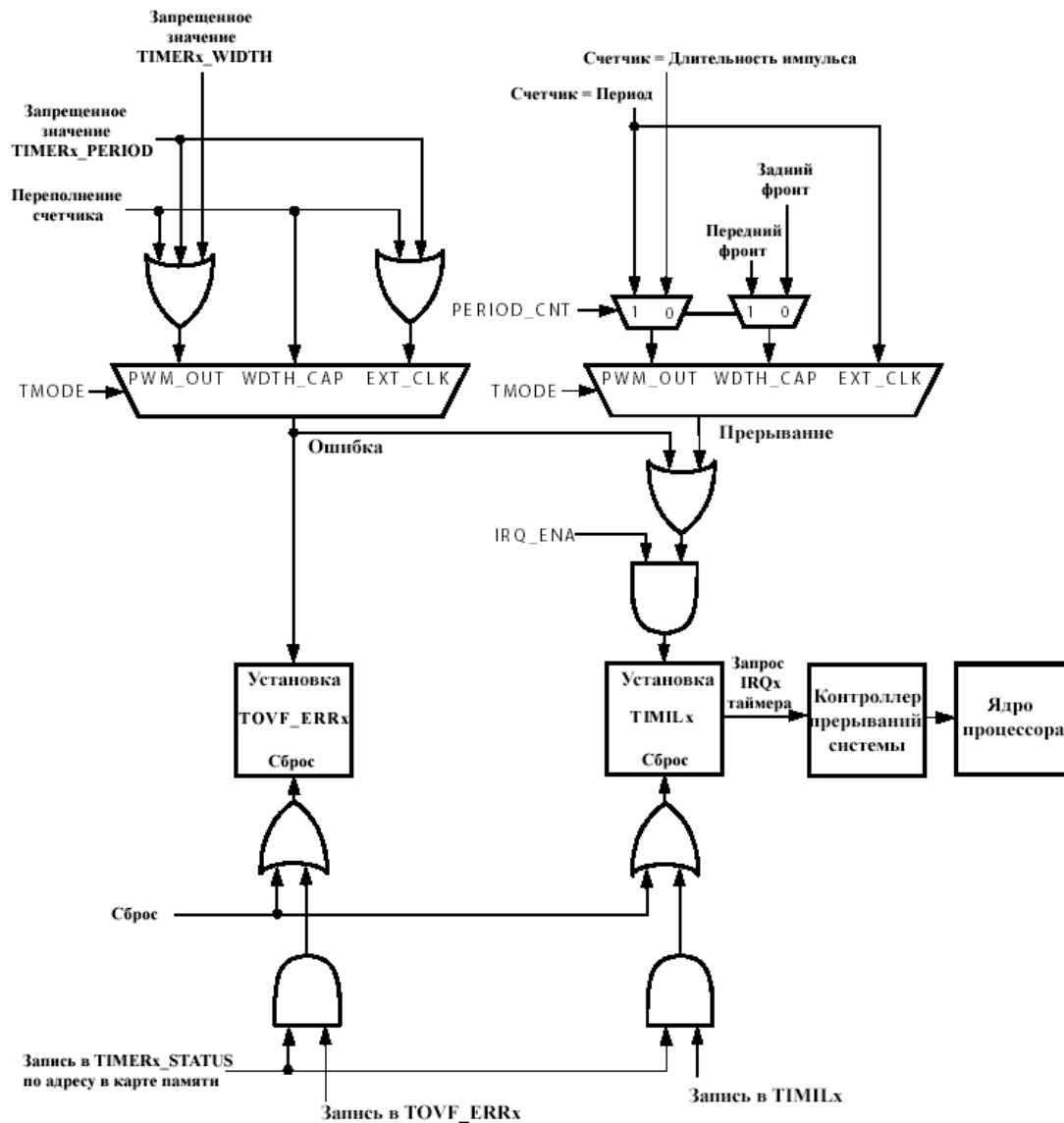


Рис. 15-24. Схема формирования прерывания таймера

Запрещённые состояния

В таблице 15-1 используются следующие определения:

- **Запуск.** Первый период тактового сигнала после включения таймера записью в `TIMER_ENABLE`.
- **Оборот счётчика.** Момент времени, в который текущее значение счётчика совпадает со значением регистра `TIMERx_PERIOD`, и происходит загрузка в счётчик значения 1.
- **Переполнение.** Ситуация, когда вместо оборота происходит инкремент счётчика, содержащего максимально возможное значение (`0xFFFF FFFF`). Диапазона значений таймера не хватает для представления следующего значения, и в него ошибочно загружается значение `0x0000 0000`.
- **Без изменения.** Новая ошибка отсутствует.

15 Таймеры

- Для ERR_TYP: поле содержит код предыдущей ошибки или 00 (если с момента разрешения работы счётчика ошибок не возникало).
- Для TOVF_ERR: если с момента разрешения работы счётчика ошибки не возникали или программа сбросила предыдущую ошибку при помощи операции W1C, при чтении возвращается 0. Если программа не подтвердила обнаружение предыдущей ошибки при чтении TOVF_ERR возвращается 1.

Для обнаружения ошибок в программе должно выполняться чтение бита TOVF_ERR. Если TOVF_ERR установлен, информацию о типе ошибки можно получить при чтении ERR_TYP. Для подтверждения обнаружения ошибки программа должна сбросить бит TOVF_ERR, выполнив операцию W1C.

В таблице 12-1 строки могут читаться следующим образом: “В режиме ____ по событию ____, если TIMERx_PERIOD = ____ и TIMERx_WIDTH = ____, то ERR_TYP = ____ и TOVF_ERR = ____”.


-  Ошибки запуска не предотвращают запуск таймера. Аналогично, работа таймера не останавливается при обороте или переполнении счётчика. Запрещенные состояния могут вызвать нежелательное поведение вывода TMRx.

Таблица 15-1. Обзор запрещённых состояний

Режим	Событие	TIMERx_PERIOD	TIMERx_WIDTH	ERR_TYP	TOVF_ERR
PWM_OUT, PERIOD_CNT = 1	Запуск (проверка граничных значений TIMERx_WIDTH не производится)	== 0	Любое значение	b#10	Установлен
		== 1	Любое значение	b#10	Установлен
		>= 2	Любое значение	Без изменения	Без изменения
	Оборот счётчика	== 0	Любое значение	b#10	Установлен
		== 1	Любое значение	b#11	Установлен
		>=2	== 0	b#11	Установлен
		>=2	< TIMERx_PERIOD	Без изменения	Без изменения
	>=2	>= TIMERx_PERIOD	b#11	Установлен	
	Переполнение (возникновение этой ошибки невозможно, если нет другой ошибки (например, TIMERx_PERIOD == 0)).	Любое значение	Любое значение	b#01	Установлен
	PWM_OUT, PERIOD_CNT = 0	Запуск	Любое значение	== 0	b#01
Любое значение			>=1	Без изменения	Без изменения
Оборот счётчика		В этом режиме оборот счётчика невозможен			

15 Таймеры

	Переполнение (возникновение этой ошибки невозможно, если нет другой ошибки (например, $TIMERx_WIDTH == 0$)).	Любое значение	Любое значение	b#01	Установлен
WDTH_CAP	Запуск	В этом режиме $TIMERx_PERIOD$ и $TIMERx_WIDTH$ доступны только для чтения, возникновение ошибки невозможно			
	Оборот счётчика	В этом режиме $TIMERx_PERIOD$ и $TIMERx_WIDTH$ доступны только для чтения, возникновение ошибки невозможно			
	Переполнение	Любое значение	Любое значение	b#01	Установлен
EXT_CLK	Запуск	$= 0$	Любое значение	b#10	Установлен
		≥ 1	Любое значение	Без изменения	Без изменения
	Оборот счётчика	$= 0$	Любое значение	b#10	Установлен
		≥ 1	Любое значение	Без изменения	Без изменения
	Переполнение (возникновение этой ошибки невозможно, если нет другой ошибки (например, $TIMERx_PERIOD == 0$)).	Любое значение	Любое значение	b#01	Установлен

Обзор

В таблице 15-2 приведён итоговый обзор использования битов и регистров управления в каждом из режимов работы таймера.

Таблица 15-2. Использование битов и регистров управления в различных режимах таймера

Бит / Регистр	Режим PWM_OUT	Режим WDTH_CAP	Режим EXT_CLK
TIMER_ENABLE	1 – Включение таймера 0 – Выключение таймера	1 – Включение таймера 0 – Не влияет на работу таймера	1 – Включение таймера 0 – Не влияет на работу таймера
TIMER_DISABLE	1 – Выключение таймера в конце периода 0 – Не влияет на работу таймера	1 – Выключение таймера 0 – Не влияет на работу таймера	1 – Выключение таймера 0 – Не влияет на работу таймера
TMODE	b#01	b#10	b#11
PULSE_HI	1 – Генерируется импульс высокого уровня 0 – Генерируется импульс низкого уровня	1 – Измеряется длительность импульса импульсы высокого уровня 0 – Измеряется длительность импульса низкого уровня	1 – Считается количество передних фронтов 0 – Считается количество задних фронтов
PERIOD_CNT	1 – Генерируется ШИМ-сигнал 0 – Генерируется одиночный импульс	1 – Прерывание генерируется по измерению периода 0 – Прерывание	Не используется

15 Таймеры

	заданной длительности	генерируется по измерению длительности импульса	
IRQ_ENA	1 – Прерывание разрешается 0 – Прерывание запрещается	1 – Прерывание разрешается 0 – Прерывание запрещается	1 – Прерывание разрешается 0 – Прерывание запрещается
TIN_SEL	Зависит от CLK_SEL: Если CLK_SEL = 1, 1 – считаются импульсы на выводе PPI_CLK 0 – считаются импульсы на выводе PF1 Если CLK_SEL = 0, не используется	1 – Выбор входа RX 0 – Выбор входа TMRx	Не используется
OUT_DIS	1 – Запрещение сигнала на выводе TMRx 0 – Разрешение сигнала на выводе TMRx	Не используется	Не используется
CLK_SEL	1 – Таймер синхронизируется сигналом PWM_CLK 0 – Таймер синхронизируется сигналом SCLK	Не используется	Не используется
TOGGLE_HI	1 – Период формируемого сигнала равен двум периодам счётчика 0 – Период формируемого сигнала равен одному периоду счётчика	Не используется	Не используется
ERR_TYP	В зависимости от типа ошибки возвращается значение b#00, b#01, b#10 или b#11	В зависимости от типа ошибки возвращается значение b#00 или b#01	В зависимости от типа ошибки возвращается значение b#00, b#01 или b#10
EMU_RUN	0 – При эмуляции таймер останавливается 1 – При эмуляции таймер работает	0 – При эмуляции таймер останавливается 1 – При эмуляции таймер работает	0 – При эмуляции таймер останавливается 1 – При эмуляции таймер работает
Вывод TMR	Зависит от OUT_DIS: 1 – находится в третьем состоянии 0 – Работает на выход	Зависит от TIN_SEL: 1 – Не используется 0 – Работает на вход	Работает на вход
Регистр периода	Доступен для чтения и записи, содержит значение периода	Доступен только для чтения, содержит значение периода	Доступен для чтения и записи, содержит значение периода
Регистр длительности импульса	Доступен для чтения и записи, содержит значение длительности импульса	Доступен только для чтения, содержит значение длительности импульса	Не используется
Регистр счётчика	Доступен только для чтения, инкрементируется по сигналу SCLK или PWM_CLK	Доступен только для чтения, инкрементируется по сигналу SCLK	Доступен только для чтения, инкрементируется по внешнему воздействию
TRUNx	При чтении: Возвращается состояние таймера (работает/остановлен) При записи: 1 – При выключении	При чтении: Возвращается состояние таймера (работает/остановлен) При записи: 1 – Не влияет на работу	При чтении: Возвращается состояние таймера (работает/остановлен) При записи: 1 – Не влияет на работу

15 Таймеры

	таймера выполняется принудительная остановка его работы 0 – Не влияет на работу таймера	таймера 0 – Не влияет на работу таймера	таймера 0 – Не влияет на работу таймера
TOVF_ERR	Устанавливается при запуске таймера или обороте счётчика, если период равен нулю или единице. Устанавливается при обороте счётчика, если длительность импульса больше или равна периоду. Устанавливается при изменении значения таймер из максимума в ноль.	Устанавливается при изменении значения таймера из максимума в ноль.	Устанавливается при изменении значения таймера из максимума в ноль. Также устанавливается при запуске таймера или обороте счётчика, если период равен нулю.
IRQ	Зависит от IRQ_ENA: 1 – Устанавливается если TOVF_ERR равен единице, когда значение счётчика равно периоду при PERIOD_CNT = 1 или когда значение счётчика равно длительности импульса при PERIOD_CNT 0 – Не устанавливается	Зависит от IRQ_ENA: 1 – Устанавливается если TOVF_ERR равен единице, когда измерен период при PERIOD_CNT = 1 или когда измерена длительность импульса при PERIOD_CNT 0 – Не устанавливается	Зависит от IRQ_ENA: 1 – Устанавливается если TOVF_ERR равен единице или когда значение счётчика равно периоду 0 – Не устанавливается

Таймер ядра

Таймер ядра – это программируемый счётчик интервалов времени, способный генерировать периодические прерывания. Таймер ядра работает с тактовой частотой системы (CCLK). Он содержит четыре регистра, отображённые в карте памяти: регистр управления таймером (TCNTL), регистр счётчика таймера (TCOUNT), регистр периода таймера (TPERIOD) и регистр масштабирования таймера (TSCALE).

На рис. 15-25 показана блок-схема таймера ядра.

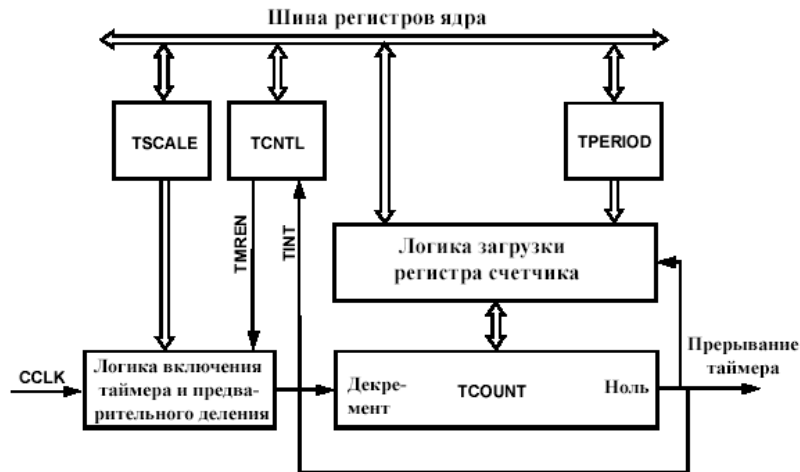


Рис. 15-25. Блок-схема таймера ядра

Регистр управления таймером ядра (TCNTL)

При включении таймера установкой бита `TMREN` в регистре `TCNTL`, содержимое регистра `TCOUNT` декрементируется через `TSCALE + 1` тактов сигнала `CCLK`. Когда значение регистра `TCOUNT` становится равным нулю, генерируется прерывание и устанавливается бит `TINT` в регистре `TCNTL`. Если в регистре `TCNTL` установлен бит `TAUTORLD`, в регистр `TCOUNT` снова загружается содержимое регистра `TPERIOD` и работа счётчика возобновляется.

Таймер ядра может быть переведён в режим низкого потребления мощности путём сброса бита `TMPCR` в регистре `TCNTL`. Перед началом использования таймера бит `TMPCR` должен быть установлен, при этом восстанавливается подача тактового сигнала на блок таймера. Когда бит `TMPCR` установлен, возможно включение таймера ядра установкой бита `TMREN` в регистре `TCNTL`.

- ⊘ Если `TMREN` устанавливается, когда `TMPCR = 0`, поведение таймера не определено.

Регистр управления таймером ядра (TCNTL)

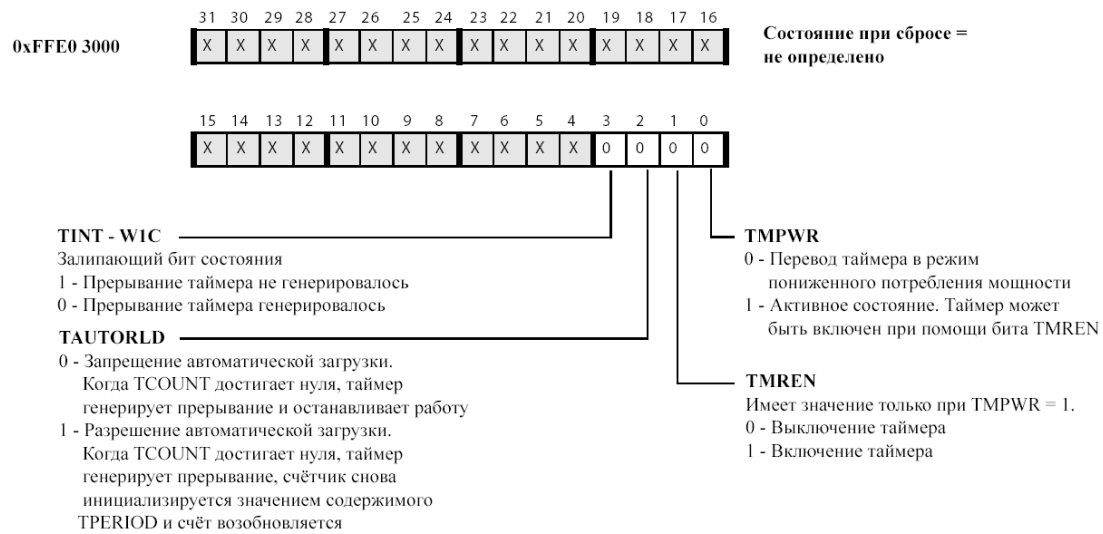


Рис.15-26. Регистр управления таймером ядра

Регистр счётчика таймера ядра (TCOUNT)

Содержимое этого регистра декрементируется через TSCALE + 1 тактов SCLK. Когда значение TCOUNT достигает нуля, генерируется прерывание и устанавливается бит TINT в регистре TCNTL.

Регистр счетчика таймера ядра (TCOUNT)

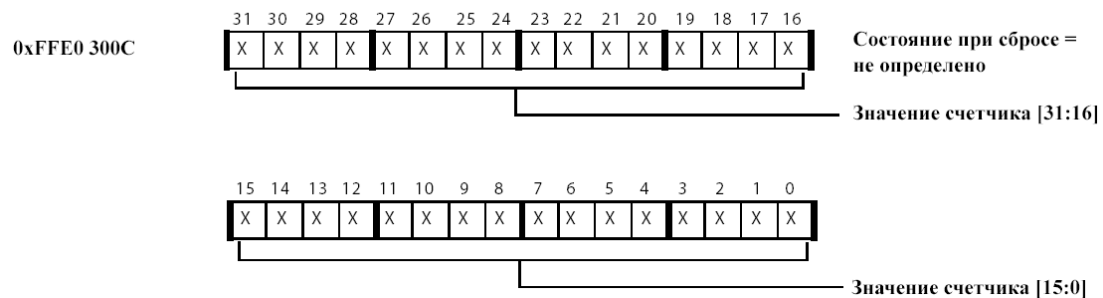


Рис. 15-27. Регистр счётчика таймера ядра

Регистр периода таймера ядра (TPERIOD)

Когда разрешена функция автоматической загрузки, каждый раз, когда значение регистра TCOUNT достигает нуля, в него загружается содержимое регистра TPERIOD.

Регистр периода таймера ядра (TPERIOD)

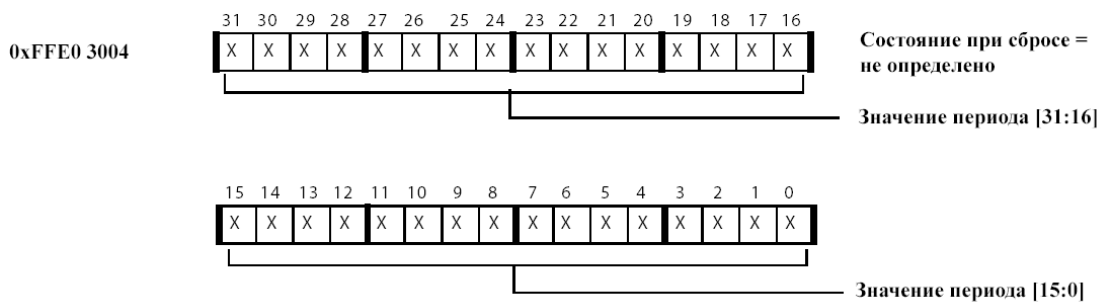


Рис. 15-28. Регистр периода таймера ядра

Регистр масштабирования таймера ядра (TSCALE)

В регистре TSCALE содержится значение коэффициента масштабирования, которое на единицу меньше, чем количество тактов между декрементом содержимого TCOUNT. Например, если значение регистра TSCALE равно 0, содержимое регистра счётчика декрементируется на каждом такте. Если TSCALE равен 1, счётчик декрементируется каждые в два такта.

Регистр масштабирования таймера ядра (TSCALE)

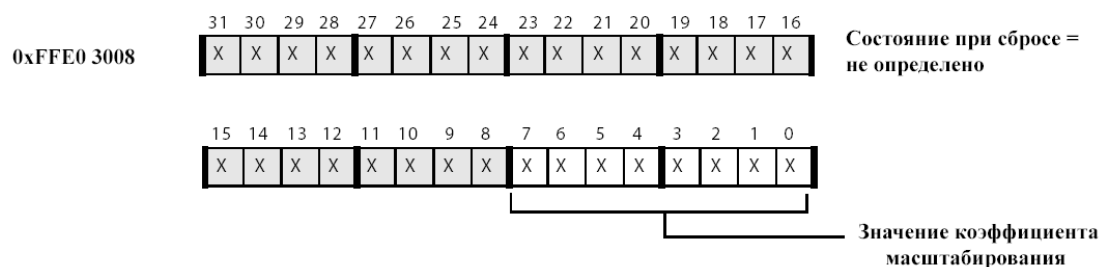


Рис. 15-29. Регистр масштабирования таймера ядра

Сторожевой таймер

Процессор имеет 32-разрядный таймер, который может использоваться для реализации функции программного сторожа. Применение программного сторожа позволяет повысить устойчивость системы при помощи генерации событий ядра процессора, если значение таймера достигает нуля до того, как он будет сброшен программно. В зависимости от настройки сторожевого таймера генерируемое событие может являться сбросом, немаскируемым прерыванием или прерыванием общего назначения. Сторожевой таймер работает от тактового сигнала системы (SCLK).

Работа сторожевого таймера

Для использования сторожевого таймера необходимо:

1. Инициализировать счётчик сторожевого таймера, записав соответствующее значение в регистр счётчика сторожевого таймера (WDOG_CNT). Следует отметить, что загрузка регистра WDOG_CNT, выполняемая до включения таймера вызывает загрузку начального значения регистра WDOG_STAT.
2. Выбрать тип события, генерируемого при таймауте, в регистре управления сторожевым таймером (WDOG_CTL).
3. Разрешить работу сторожевого таймера в регистре WDOG_CTL. При этом сторожевой таймер начинает счёт вниз, декрементируя содержимое регистра WDOG_STAT. Когда значение WDOG_STAT достигает нуля, генерируется заданное событие. Для того чтобы событие не генерировалось, программа должна до того, как счётчик достигнет нуля, выполнить загрузку содержимого WDOG_CNT в WDOG_STAT при помощи записи (любого значения) в WDOG_STAT или выключить сторожевой таймер.

Регистр счётчика сторожевого таймера (WDOG_CNT)

Регистр счётчика сторожевого таймера (WDOG_CNT) содержит 32-разрядное беззнаковое начальное значение счётчика. Обращение к регистру WDOG_CNT должно выполняться при помощи 32-разрядных операций чтения/записи.

Регистр счётчика сторожевого таймера содержит задаваемое начальное значение счётчика. При корректной записи в регистр счётчика сторожевого таймера выполняется инициализация счётчика сторожевого таймера. В целях дополнительной защиты обновление регистра счётчика сторожевого таймера возможно только при выключенном сторожевом таймере. Запись в регистр счётчика сторожевого таймера во время работы таймера не изменяет содержимого регистра.

Регистр счетчика сторожевого таймера (WDOG_CNT)

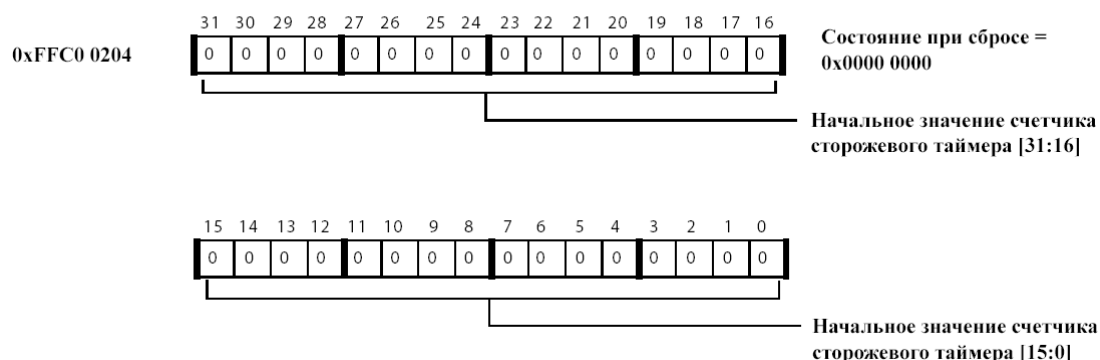


Рис. 15-30. Регистр счётчика сторожевого таймера

Регистр состояния сторожевого таймера (WDOG_STAT)

32-разрядный регистр состояния сторожевого таймера (WDOG_STAT) содержит текущее значение счётчика сторожевого таймера. При чтении регистра WDOG_STAT возвращается текущее значение счётчика. Во время работы сторожевого таймера содержимое WDOG_STAT декрементируется на единицу на каждом такте SCLK. Когда значение WDOG_STAT достигает нуля, счёт сторожевого таймера прекращается и генерируется заданное в регистре управления сторожевым таймером (WDOG_CTL) событие.

Прямая запись значений в регистр WDOG_STAT невозможна. В него может копироваться содержимое регистра WDOG_CNT двумя способами:

- Когда сторожевой таймер выключен, запись в регистр WDOG_CNT вызывает инициализацию регистра WDOG_STAT.
- Когда сторожевой таймер работает, запись значения в регистр WDOG_STAT приводит к загрузке в него содержимого регистра WDOG_CNT.

При выполнении процессором записи произвольного значения в регистр WDOG_STAT, в него копируется содержимое регистра WDOG_CNT. Обычно в программе при инициализации задаётся значение регистра WDOG_CNT, а затем, периодически, пока счётчик не достиг нуля, выполняется запись в регистр WDOG_STAT. Это приводит к загрузке в сторожевой таймер содержимого WDOG_CNT и предотвращает генерацию выбранного события.

Регистр WDOG_STAT является 32-разрядным беззнаковым регистром, отображённым в карте памяти. Обращение к нему должно выполняться при помощи 32-разрядных операций чтения/записи.

Если пользователь не перезагружает счётчик в течение $SCLK \times WDOG_CNT$ тактов, генерируется сброс или прерывание сторожевого таймера и устанавливается бит TRO в регистре управления сторожевым таймером. При этом прекращается декремент счётчика, и значение счётчика остаётся равным нулю.

Если при включении таймера в счётчик загружается ноль, незамедлительно устанавливается бит TRO в регистре управления сторожевым таймером. При этом счётчик не декрементируется, и значение счётчика остаётся равным нулю.

Регистр состояния сторожевого таймера (WDOG_STAT)

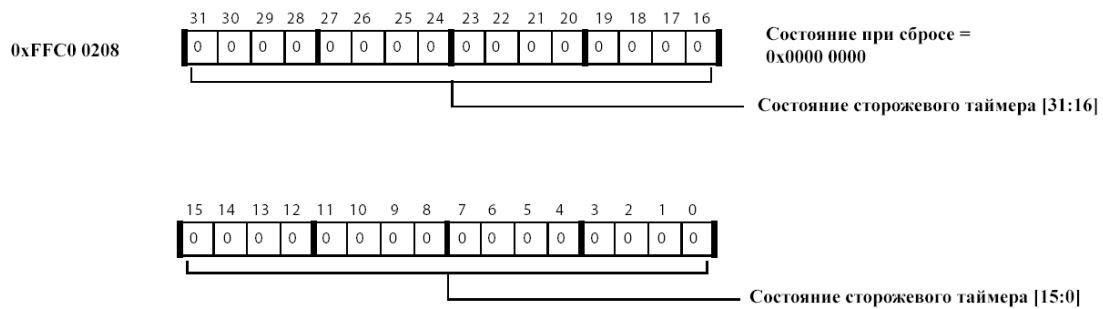


Рис. 15-31. Регистр состояния сторожевого таймера

Регистр управления сторожевым таймером (WDOG_CTL)

Регистр управления сторожевым таймером (WDOG_CTL) представляет собой 16-разрядный регистр системы, отображённый в карте памяти и используемый для управления сторожевым таймером.

Поле ICTL[1:0] используется для выбора события, генерируемого по истечению сторожевого таймера. Необходимо отметить, что если выбрана генерация прерывания общего назначения, маскирование прерывания должно быть снято в регистре маскирования прерываний системы (SIC_IMASK). Если генерация прерывания сторожевым таймером запрещена, он работает в соответствии с процедурой, описанной выше, за исключением того, что по достижении счётчиком нуля прерывание не генерируется.

Поле TMR_EN[7:0] используется для включения и выключения сторожевого таймера. Запись в это поле любого значения, за исключением кода выключения таймера, запускает работу таймера. Использование многоразрядного кода выключения минимизирует вероятность непреднамеренного выключения сторожевого таймера.

Программа может определить, достиг ли таймер нуля, путём опроса бита состояния TRO регистра управления сторожевым таймером. Этот бит является залипающим и устанавливается каждый раз, когда значение счётчика сторожевого таймера достигает нуля; сброс бита выполняется только при записи в него единицы.

Регистр управления сторожевым таймером (WDOG_CTL)

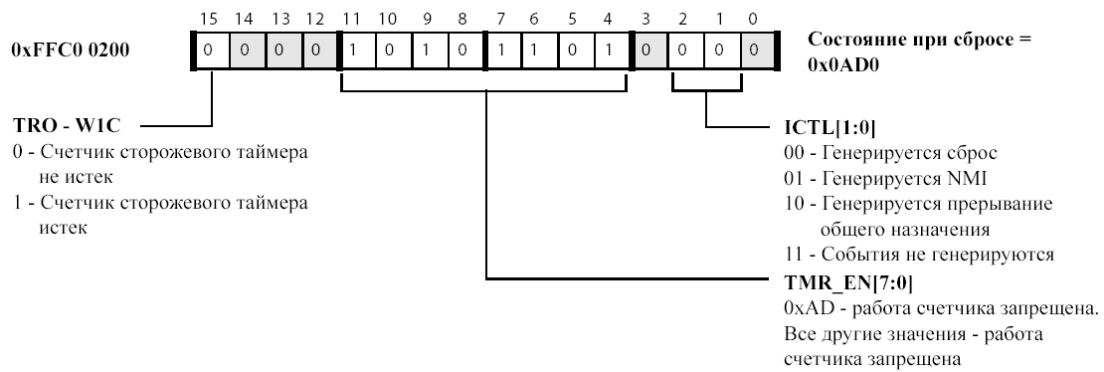


Рис. 15-32. Регистр управления сторожевым таймером

i Необходимо отметить, что в режиме Эмуляции счётчик сторожевого таймера не декрементируется, даже если он включён.