# Описание используемых типов

## TIMER\_CntInit\_type

|  |  |
| --- | --- |
| **Имя поля** | **Описание** |
| IniCounter | Начальное значение счетчика |
| Prescaler | Делитель тактовой частоты |
| Period | Основание счета (период) |
| CounterMode | Режим счета:  *TIMER\_CntMode\_ClkFixedDir* – счет без изменения направления от внутреннего тактового сигнала  *TIMER\_CntMode\_ClkChangeDir* – счет с изменением направления от внутреннего тактового сигнала  *TIMER\_CntMode\_EvtFixedDir* – счет без изменения направления от событий  *TIMER\_CntMode\_EvtChangeDir* – счет с изменением направления от событий |
| CounterDirection | Направление счета:  *TIMER\_CntDir\_Up* – счет вверх  *TIMER\_CntDir\_Down –* счет вниз |
| EventSource | Источник событий:  *TIMER\_EvSrc\_None* ­– нет событий  *TIMER\_EvSrc\_TM1* – CNT == ARR в таймере 1  *TIMER\_EvSrc\_TM2* – CNT == ARR в таймере 2  *TIMER\_EvSrc\_CH1* – событие на первом канале  *TIMER\_EvSrc\_CH2* – событие на втором канале  *TIMER\_EvSrc\_CH3* – событие на третьем канале  *TIMER\_EvSrc\_CH4* – событие на четвертом канале  *TIMER\_EvSrc\_ETR\_RE* – событие переднего фронта ETR  *TIMER\_EvSrc\_ETR\_FE* – событие заднего фронта ETR |
| FilterSampling | Частота семплирования данных:  *TIMER\_FdtsDiv\_1* – частота семплирования каждый TIM\_CLK  *TIMER\_FdtsDiv\_2* – частота семплирования каждый второй TIM\_CLK  *TIMER\_FdtsDiv\_3* – частота семплирования каждый третий TIM\_CLK  *TIMER\_FdtsDiv\_4* – частота семплирования каждый четвертый TIM\_CLK |
| ARR\_UpdateMode | Режим обновления регистра ARR:  *TIMER\_Update\_Immediately* – обновление возможно в любой момент времени  *TIMER\_Update\_On\_CNT\_Overflow* – обновление будет осуществлено только при CNT = 0 |
| ETR\_FilterConf | Настройка фильтра на входе ETR:  *TIMER\_Filter\_1FF\_at\_TIMER\_CLK* – сигнал зафиксирован в 1 триггере на частоте TIM\_CLK  *TIMER\_Filter\_2FF\_at\_TIMER\_CLK* – сигнал зафиксирован в 2 триггерах на частоте TIM\_CLK  *TIMER\_Filter\_4FF\_at\_TIMER\_CLK* – сигнал зафиксирован в 4 триггерах на частоте TIM\_CLK  *TIMER\_Filter\_8FF\_at\_TIMER\_CLK* – сигнал зафиксирован в 8 триггерах на частоте TIM\_CLK  *TIMER\_Filter\_6FF\_at\_FDTS\_div\_2* – сигнал зафиксирован в 6 триггерах на частоте FDTS/2  *TIMER\_Filter\_8FF\_at\_FDTS\_div\_2* – сигнал зафиксирован в 8 триггерах на частоте FDTS/2  *TIMER\_Filter\_6FF\_at\_FDTS\_div\_4* – сигнал зафиксирован в 6 триггерах на частоте FDTS/4  *TIMER\_Filter\_8FF\_at\_FDTS\_div\_4* – сигнал зафиксирован в 8 триггерах на частоте FDTS/4  *TIMER\_Filter\_6FF\_at\_FDTS\_div\_8* – сигнал зафиксирован в 6 триггерах на частоте FDTS/8  *TIMER\_Filter\_8FF\_at\_FDTS\_div\_8* –сигнал зафиксирован в 8 триггерах на частоте FDTS/8  *TIMER\_Filter\_5FF\_at\_FDTS\_div\_16* – сигнал зафиксирован в 5 триггерах на частоте FDTS/16  *TIMER\_Filter\_6FF\_at\_FDTS\_div\_16* – сигнал зафиксирован в 6 триггерах на частоте FDTS/16  *TIMER\_Filter\_8FF\_at\_FDTS\_div\_16* – сигнал зафиксирован в 8 триггерах на частоте FDTS/16  *TIMER\_Filter\_5FF\_at\_FDTS\_div\_32* – сигнал зафиксирован в 5 триггерах на частоте FDTS/32  *TIMER\_Filter\_6FF\_at\_FDTS\_div\_32* – сигнал зафиксирован в 6 триггерах на частоте FDTS/32  *TIMER\_Filter\_8FF\_at\_FDTS\_div\_32* – сигнал зафиксирован в 8 триггерах на частоте FDTS/32 |
| ETR\_Prescaler | Асинхронный пред делитель внешней частоты:  *TIMER\_Prescaler\_None* – нет деления  *TIMER\_Prescaler\_2* – /2  *TIMER\_Prescaler\_4* – /4  *TIMER\_Prescaler\_8* – /8 |
| ETR\_Polarity | Инверсия ETR:  *TIMER\_Polarity\_NonInverted* – нет инверсии входа  *TIMER\_Polarity\_Inverted* – инверсия входа |
| BRK\_Polarity | Инверсия BRK:  *TIMER\_Polarity\_NonInverted* – нет инверсии входа  *TIMER\_Polarity\_Inverted* – инверсия входа |

## TIMER\_ChnInit\_type

|  |  |
| --- | --- |
| **Имя поля** | **Описание** |
| ChNumber | Номер канала таймера:  *TIMER\_CH1*  *TIMER\_CH2*  *TIMER\_CH3*  *TIMER\_CH4* |
| Mode | Режим работы канала:  *TIMER\_CH\_MODE\_PWM* – канал работает в режиме ШИМ  *TIMER\_CH\_MODE\_CAPTURE* – канал работает в режиме Захват |
| ETR\_Ena | Разрешение работы ETR:  *DISABLE*  *ENABLE* |
| ETR\_Reset | Разрешение сброса по выводу ETR:  *TIMER\_CH\_RESET\_DISABLE* – сброс по внешнему выводу запрещен  *TIMER\_CH\_RESET\_ENABLE* – сброс по внешнему выводу разрешен |
| BRK\_Reset | Разрешение сброса по выводу BRK:  *TIMER\_CH\_RESET\_DISABLE* – сброс по внешнему выводу запрещен  *TIMER\_CH\_RESET\_ENABLE* – сброс по внешнему выводу разрешен |
| REF\_Format | Формат выработки сигнала REF в режиме ШИМ:  *TIMER\_CH\_REF\_Format0* – REF = 0  *TIMER\_CH\_REF\_Format1* – REF = 1 если: (CNT == CCR) или ((CNT == CCR1) и CCR1\_EN = 1), иначе REF = 0  *TIMER\_CH\_REF\_Format2* – REF = 0 если: (CNT == CCR) или ((CNT == CCR1) и CCR1\_EN = 1), иначе REF = 1  *TIMER\_CH\_REF\_Format3* – Переключение REF если: (CNT == CCR) или ((CNT == CCR1) и CCR1\_EN = 1)  *TIMER\_CH\_REF\_Format4* – REF = 0  *TIMER\_CH\_REF\_Format5* – REF = 1  *TIMER\_CH\_REF\_Format6* – REF = ~DIR если: (CCR1\_EN = 0 и (CNT < CCR)) или (CCR1\_EN = 1 и (CCR1 < CNT < CCR)), иначе REF = DIR  *TIMER\_CH\_REF\_Format7* – REF = DIR если: (CCR1\_EN = 0 и (CNT < CCR)) или (CCR1\_EN = 1 и (CCR1 < CNT < CCR)), иначе REF = ~DIR |
| Prescaler | Предварительный делитель входного канала:  *TIMER\_Prescaler\_None* – нет деления  *TIMER\_Prescaler\_2* – /2  *TIMER\_Prescaler\_4* – /4  *TIMER\_Prescaler\_8* – /8 |
| EventSource | Выбор события по входному каналу:  *TIMER\_EventIn\_PE* – Событие входного канала положительный фронт  *TIMER\_EventIn\_NE* – Событие входного канала отрицательный фронт  *TIMER\_EventIn\_EX1* – Событие входного канала положительный фронт от других каналов ( 1<-2, 2<-3, 3<-4, 4<-1 )  *TIMER\_EventIn\_EX2* – Событие входного канала положительный фронт от других каналов ( 1<-3, 2<-4, 3<-1, 4<-2 ) |
| FilterConf | Фильтрация канала:  *TIMER\_Filter\_1FF\_at\_TIMER\_CLK* – сигнал зафиксирован в 1 триггере на частоте TIM\_CLK  *TIMER\_Filter\_2FF\_at\_TIMER\_CLK* – сигнал зафиксирован в 2 триггерах на частоте TIM\_CLK  *TIMER\_Filter\_4FF\_at\_TIMER\_CLK* – сигнал зафиксирован в 4 триггерах на частоте TIM\_CLK  *TIMER\_Filter\_8FF\_at\_TIMER\_CLK* – сигнал зафиксирован в 8 триггерах на частоте TIM\_CLK  *TIMER\_Filter\_6FF\_at\_FDTS\_div\_2* – сигнал зафиксирован в 6 триггерах на частоте FDTS/2  *TIMER\_Filter\_8FF\_at\_FDTS\_div\_2* – сигнал зафиксирован в 8 триггерах на частоте FDTS/2  *TIMER\_Filter\_6FF\_at\_FDTS\_div\_4* – сигнал зафиксирован в 6 триггерах на частоте FDTS/4  *TIMER\_Filter\_8FF\_at\_FDTS\_div\_4* – сигнал зафиксирован в 8 триггерах на частоте FDTS/4  *TIMER\_Filter\_6FF\_at\_FDTS\_div\_8* – сигнал зафиксирован в 6 триггерах на частоте FDTS/8  *TIMER\_Filter\_8FF\_at\_FDTS\_div\_8* –сигнал зафиксирован в 8 триггерах на частоте FDTS/8  *TIMER\_Filter\_5FF\_at\_FDTS\_div\_16* – сигнал зафиксирован в 5 триггерах на частоте FDTS/16  *TIMER\_Filter\_6FF\_at\_FDTS\_div\_16* – сигнал зафиксирован в 6 триггерах на частоте FDTS/16  *TIMER\_Filter\_8FF\_at\_FDTS\_div\_16* – сигнал зафиксирован в 8 триггерах на частоте FDTS/16  *TIMER\_Filter\_5FF\_at\_FDTS\_div\_32* – сигнал зафиксирован в 5 триггерах на частоте FDTS/32  *TIMER\_Filter\_6FF\_at\_FDTS\_div\_32* – сигнал зафиксирован в 6 триггерах на частоте FDTS/32  *TIMER\_Filter\_8FF\_at\_FDTS\_div\_32* – сигнал зафиксирован в 8 триггерах на частоте FDTS/32 |
| CCR\_UpdateMode | Настройка обновления регистров CCR и CCR1:  *TIMER\_Update\_Immediately* – обновление возможно в любой момент времени  *TIMER\_Update\_On\_CNT\_Overflow* – обновление будет осуществлено только при CNT = 0 |
| CCR1\_Ena | Разрешение работы регистра CCR1:  *DISABLE*  *ENABLE* |
| CCR1\_EventSource | Выбор события по входному каналу для CAP1:  *TIMER\_EventIn\_PE* – событие входного канала положительный фронт  *TIMER\_EventIn\_NE* – событие входного канала отрицательный фронт  *TIMER\_EventIn\_EX1* – событие входного канала положительный фронт от других каналов ( 1<-2, 2<-3, 3<-4, 4<-1 )  *TIMER\_EventIn\_EX2* – событие входного канала положительный фронт от других каналов ( 1<-3, 2<-4, 3<-1, 4<-2 ) |

## TIMER\_ChnOutInit\_type

|  |  |
| --- | --- |
| **Имя поля** | **Описание** |
| ChNumber | Номер канала таймера:  *TIMER\_CH1*  *TIMER\_CH2*  *TIMER\_CH3*  *TIMER\_CH4* |
| DirOut\_Polarity | Инверсия прямого канала:  *TIMER\_CHOPolarity\_NonInverted* – выход не инвертируется  *TIMER\_CHOPolarity\_Inverted* – выход не инвертируется |
| DirOut\_Source | Источник сигнала прямого канала:  *TIMER\_CH\_OutSrc\_Only\_0* – На выход подается всегда 0  *TIMER\_CH\_OutSrc\_Only\_1* – На выход подается всегда 1  *TIMER\_CH\_OutSrc\_REF* – На выход выдается сигнал REF  *TIMER\_CH\_OutSrc\_DTG* – На выход выдается сигнал с DTG |
| DirOut\_Mode | Режим работы прямого канала:  *TIMER\_CH\_OutMode\_Input* – Канал работает на вход  *TIMER\_CH\_OutMode\_Output* – Канал всегда работает на выход  *TIMER\_CH\_OutMode\_REF\_as\_OE* – Режим канала зависит от сигнала REF  *TIMER\_CH\_OutMode\_DTG\_as\_OE* – Режим канала зависит от сигнала DTG |
| NegOut\_Polarity | Инверсия инверсного канала:  *TIMER\_CHOPolarity\_NonInverted* – выход не инвертируется  *TIMER\_CHOPolarity\_Inverted* – выход не инвертируется |
| NegOut\_Source | Источник сигнала инверсного канала:  *TIMER\_CH\_OutSrc\_Only\_0* – На выход подается всегда 0  *TIMER\_CH\_OutSrc\_Only\_1* – На выход подается всегда 1  *TIMER\_CH\_OutSrc\_REF* – На выход выдается сигнал REF  *TIMER\_CH\_OutSrc\_DTG* – На выход выдается сигнал с DTG |
| NegOut\_Mode | Режим работы инверсного канала:  *TIMER\_CH\_OutMode\_Input* – Канал работает на вход  *TIMER\_CH\_OutMode\_Output* – Канал всегда работает на выход  *TIMER\_CH\_OutMode\_REF\_as\_OE* – Режим канала зависит от сигнала REF  *TIMER\_CH\_OutMode\_DTG\_as\_OE* – Режим канала зависит от сигнала DTG |
| DTG\_MainPrescaler | Основной делитель частоты генератора мертвого времени |
| DTG\_AuxPrescaler | Предварительный делитель частоты генератора мертвого времени |
| DTG\_ClockSource | Источник тактирования генератора мертвой времени:  *TIMER\_CH\_DTG\_ClkSrc\_TIMER\_CLK* – Частота TIM\_CLK  *TIMER\_CH\_DTG\_ClkSrc\_FDTS* – Частота FDTS |

# Описание используемых функций

## HAL\_TIMER\_DeInit

**Резюме**

**void HAL\_TIMER\_DeInit**( TIMER\_type\* TIMERx )

TIMER\_type\* TIMERx – указатель на структуру с регистрами таймера

**Описание**

Функция сбрасывает периферию выбранного таймера – записывает нули в регистры.

## HAL\_TIMER\_CntInit

**Резюме**

**void** **HAL\_TIMER\_CntInit**( TIMER\_type\* TIMERx, **const** TIMER\_CntInit\_type\* TIMER\_CntInitStruct )

TIMER\_type\* TIMERx – указатель на структуру с регистрами таймера

**const** TIMER\_CntInit\_type\* TIMER\_CntInitStruct – указатель на структуру с параметрами инициализации

**Описание**

Функция инициализирует регистры отвечающие за режим счета таймера параметрами из инициализационной структуры.

## HAL\_TIMER\_CntStructInit

**Резюме**

**void** **HAL\_TIMER\_CntStructInit**( TIMER\_CntInit\_type\*   
TIMER\_CntInitStruct )

TIMER\_CntInit\_type\* TIMER\_CntInitStruct – указатель на структуру с параметрами инициализации

**Описание**

Функция задает параметры инициализационной структуры значениями по умолчанию:

TIMER\_CntInitStruct->IniCounter = 0;

TIMER\_CntInitStruct->Prescaler = 0;

TIMER\_CntInitStruct->Period = 0;

TIMER\_CntInitStruct->CounterMode = *TIMER\_CntMode\_ClkFixedDir*;

TIMER\_CntInitStruct->CounterDirection = *TIMER\_CntDir\_Up*;

TIMER\_CntInitStruct->EventSource = *TIMER\_EvSrc\_None*;

TIMER\_CntInitStruct->FilterSampling = *TIMER\_FdtsDiv\_1*;

TIMER\_CntInitStruct->ARR\_UpdateMode = *TIMER\_Update\_Immediately*;

TIMER\_CntInitStruct->ETR\_FilterConf = *TIMER\_Filter\_1FF\_at\_TIMER\_CLK*;

TIMER\_CntInitStruct->ETR\_Prescaler = *TIMER\_Prescaler\_None*;

TIMER\_CntInitStruct->ETR\_Polarity = *TIMER\_Polarity\_NonInverted*;

TIMER\_CntInitStruct->BRK\_Polarity = *TIMER\_Polarity\_NonInverted*;

## HAL\_TIMER\_Cmd

**Резюме**

**void** **HAL\_TIMER\_Cmd**( TIMER\_type\* TIMERx, FunctionalState\_type NewState )

TIMER\_type\* TIMERx – указатель на структуру с регистрами таймера

FunctionalState\_type NewState – состояние (*ENABLE, DISABLE*)

**Описание**

Функция включат или выключает выбранный таймер.

**HAL\_TIMER\_SetCounter**

**Резюме**

**void** **HAL\_TIMER\_SetCounter**( TIMER\_type\* TIMERx, uint32\_t Counter )

TIMER\_type\* TIMERx – указатель на структуру с регистрами таймера

uint32\_t Counter – значение счетчика

**Описание**

Функция задает значение счетчика (регистр CNT) выбранного таймера.

## HAL\_TIMER\_SetCntPrescaler

**Резюме**

**void HAL\_TIMER\_SetCntPrescaler**( TIMER\_type\* TIMERx, uint16\_t Prescaler )

TIMER\_type\* TIMERx – указатель на структуру с регистрами таймера

uint16\_t Prescaler – значение делителя частоты

**Описание**

Функция задает значение делителя частоты тактирования (регистр PSG) выбранного таймера.

## HAL\_TIMER\_CntAutoreloadConfig

**Резюме**

**void** **HAL\_TIMER\_CntAutoreloadConfig**( TIMER\_type\* TIMERx, uint32\_t Autoreload, TIMER\_Update\_type UpdateMode )

TIMER\_type\* TIMERx – указатель на структуру с регистрами таймера

uint32\_t Autoreload – значение при котором происходит обнуление счетчика

TIMER\_Update\_type UpdateMode – когда загружать новое значение (сразу – *TIMER\_Update\_Immediately*, при CNT = 0 – *TIMER\_Update\_On\_CNT\_Overflow*)

**Описание**

Функция задает значение регистра основания счета (регистр ARR) выбранного таймера и момент его обновления.

## HAL\_TIMER\_GetCounter

**Резюме**

uint32\_t **HAL\_TIMER\_GetCounter**( TIMER\_type\* TIMERx )

TIMER\_type\* TIMERx – указатель на структуру с регистрами таймера

**Описание**

Функция возвращает текущее значение счетного регистра CNT выбранного таймера.

## HAL\_TIMER\_GetCounterDirection

**Резюме**

TIMER\_CNTRL\_DIR\_type **HAL\_TIMER\_GetCounterDirection**( TIMER\_type\* TIMERx )

TIMER\_type\* TIMERx – указатель на структуру с регистрами таймера

**Описание**

Функция возвращает текущее значение направления счета выбранного таймера:

*TIMER\_CntDir\_Up* – счет вверх

*TIMER\_CntDir\_Down* – счет вниз

## HAL\_TIMER\_GetCntWriteComplete

**Резюме**

State\_type **HAL\_TIMER\_GetCntWriteComplete**( TIMER\_type\* TIMERx )

TIMER\_type\* TIMERx – указатель на структуру с регистрами таймера

**Описание**

Функция возвращает текущее состояние записи регистров CNT, PSG и ARR выбранного таймера:

*RESET* = 0

*SET* = 1

## HAL\_TIMER\_ETRInputConfig

**Резюме**

**void** **HAL\_TIMER\_ETRInputConfig**( TIMER\_type\* TIMERx, TIMER\_Prescaler\_type Prescaler, TIMER\_Polarity\_type Polarity, TIMER\_Filter\_type Filter )

TIMER\_type\* TIMERx – указатель на структуру с регистрами таймера

TIMER\_Prescaler\_type Prescaler – предделитель

TIMER\_Polarity\_type Polarity – настройка инверсии

TIMER\_Filter\_type Filter – цифровой фильтр

**Описание**

Функция настраивает параметры входа внешнего тактирования ETR выбранного таймера. Значения аргументов описаны в структуре TIMER\_ChnInit\_type и соответствуют полям ETR\_Prescaler, ETR\_Polarity, ETR\_FilterConf.

## HAL\_TIMER\_BRKPolarityConfig

**Резюме**

**void** **HAL\_TIMER\_BRKPolarityConfig**( TIMER\_type\* TIMERx, TIMER\_Polarity\_type Polarity )

TIMER\_type\* TIMERx – указатель на структуру с регистрами таймера

TIMER\_Polarity\_type Polarity – настройка инверсии

**Описание**

Функция настраивает параметры входа внешнего сигнала сброса BRK выбранного таймера. Значение аргумента описаны в структуре TIMER\_ChnInit\_type и соответствуют полю BRK\_Polarity.

## HAL\_TIMER\_ChnInit

**Резюме**

**void** **HAL\_TIMER\_ChnInit**( TIMER\_type\* TIMERx, **const** TIMER\_ChnInit\_type\* TIMER\_ChnInitStruct )

TIMER\_type\* TIMERx – указатель на структуру с регистрами таймера

**const** TIMER\_ChnInit\_type\* TIMER\_ChnInitStruct – указатель на структуру с параметрами инициализации канала таймера

**Описание**

Функция инициализирует регистры отвечающие за режим работы канала таймера параметрами из инициализационной структуры.

## HAL\_TIMER\_ChnStructInit

**Резюме**

**void** **HAL\_TIMER\_ChnStructInit**( TIMER\_ChnInit\_type\*   
TIMER\_ChnInitStruct )

TIMER\_ChnInit\_type\* TIMER\_ChnInitStruct – указатель на структуру с параметрами инициализации канала таймера

**Описание**

Функция задает параметры инициализационной структуры канала таймера значениями по умолчанию:

TIMER\_ChnInitStruct->ChNumber = *TIMER\_CH1*;

TIMER\_ChnInitStruct->Mode = *TIMER\_CH\_MODE\_PWM*;

TIMER\_ChnInitStruct->ETR\_Ena = *DISABLE*;

TIMER\_ChnInitStruct->ETR\_Reset = *TIMER\_CH\_RESET\_DISABLE*;

TIMER\_ChnInitStruct->BRK\_Reset = *TIMER\_CH\_RESET\_DISABLE*;

TIMER\_ChnInitStruct->REF\_Format = *TIMER\_CH\_REF\_Format0*;

TIMER\_ChnInitStruct->Prescaler = *TIMER\_Prescaler\_None*;

TIMER\_ChnInitStruct->EventSource = *TIMER\_EventIn\_PE*;

TIMER\_ChnInitStruct->FilterConf = *TIMER\_Filter\_1FF\_at\_TIMER\_CLK*;

TIMER\_ChnInitStruct->CCR\_UpdateMode = *TIMER\_Update\_Immediately*;

TIMER\_ChnInitStruct->CCR1\_Ena = *DISABLE*;

TIMER\_ChnInitStruct->CCR1\_EventSource = *TIMER\_EventIn\_PE*;

## HAL\_TIMER\_ChnCompareSet

**Резюме**

**void** **HAL\_TIMER\_ChnCompareSet**( TIMER\_type\* TIMERx, TIMER\_CH\_type Channel, uint32\_t Compare )

TIMER\_type\* TIMERx – указатель на структуру с регистрами таймера

TIMER\_CH\_type Channel – канала таймера (*TIMER\_CH1 – TIMER\_CH4*)

uint32\_t Compare – значение регистра сравнения

**Описание**

Функция задает значение регистра сравнения/захвата канала у выбранного таймера.

## HAL\_TIMER\_ChnCompareConfig

**Резюме**

**void** **HAL\_TIMER\_ChnCompareConfig**( TIMER\_type\* TIMERx, TIMER\_CH\_type Channel, uint32\_t Compare, TIMER\_Update\_type UpdateMode )

TIMER\_type\* TIMERx – указатель на структуру с регистрами таймера

TIMER\_CH\_type Channel – канала таймера (*TIMER\_CH1 – TIMER\_CH4*)

uint32\_t Compare – значение регистра сравнения

TIMER\_Update\_type UpdateMode – когда загружать новое значение (сразу – *TIMER\_Update\_Immediately*, при CNT = 0 – *TIMER\_Update\_On\_CNT\_Overflow*)

**Описание**

Функция задает значение регистра сравнения/захвата канала у выбранного таймера, с заданием момента его обновления.

## HAL\_TIMER\_ChnCompare1Set

**Резюме**

**void HAL\_TIMER\_ChnCompare1Set**( TIMER\_type\* TIMERx, TIMER\_CH\_type Channel, uint32\_t Compare )

TIMER\_type\* TIMERx – указатель на структуру с регистрами таймера

TIMER\_CH\_type Channel – канала таймера (*TIMER\_CH1 – TIMER\_CH4*)

uint32\_t Compare – значение регистра сравнения

**Описание**

Функция задает значение регистра сравнения/захвата один канала у выбранного таймера.

## HAL\_TIMER\_ChnCompare1Config

**Резюме**

**void** **HAL\_TIMER\_ChnCompare1Config**( TIMER\_type\* TIMERx, TIMER\_CH\_type Channel, uint32\_t Compare, TIMER\_Update\_type UpdateMode )

TIMER\_type\* TIMERx – указатель на структуру с регистрами таймера

TIMER\_CH\_type Channel – канала таймера (*TIMER\_CH1 – TIMER\_CH4*)

uint32\_t Compare – значение регистра сравнения

TIMER\_Update\_type UpdateMode – когда загружать новое значение (сразу – *TIMER\_Update\_Immediately*, при CNT = 0 – *TIMER\_Update\_On\_CNT\_Overflow*)

**Описание**

Функция задает значение регистра сравнения/захвата один канала у выбранного таймера, с заданием момента его обновления.

## HAL\_TIMER\_GetChnCapture

**Резюме**

uint32\_t **HAL\_TIMER\_GetChnCapture**( TIMER\_type\* TIMERx, TIMER\_CH\_type Channel )

TIMER\_type\* TIMERx – указатель на структуру с регистрами таймера

TIMER\_CH\_type Channel – канала таймера (*TIMER\_CH1 – TIMER\_CH4*)

**Описание**

Функция возвращает текущее значение регистра сравнения/захвата канала у выбранного таймера.

## HAL\_TIMER\_GetChnCapture1

**Резюме**

uint32\_t **HAL\_TIMER\_GetChnCapture1**( TIMER\_type\* TIMERx, TIMER\_CH\_type Channel )

TIMER\_type\* TIMERx – указатель на структуру с регистрами таймера

TIMER\_CH\_type Channel – канала таймера (*TIMER\_CH1 – TIMER\_CH4*)

**Описание**

Функция возвращает текущее значение регистра сравнения/захвата один канала у выбранного таймера.

## HAL\_TIMER\_ChnETR\_Cmd

**Резюме**

**void** **HAL\_TIMER\_ChnETR\_Cmd**( TIMER\_type\* TIMERx, TIMER\_CH\_type Channel, FunctionalState\_type NewState )

TIMER\_type\* TIMERx – указатель на структуру с регистрами таймера

TIMER\_CH\_type Channel – канала таймера (*TIMER\_CH1 – TIMER\_CH4*)

FunctionalState\_type NewState – состояние (*ENABLE, DISABLE*)

**Описание**

Функция управления выводом входа внешней тактовой частоты ETR.

## HAL\_TIMER\_ChnETRResetConfig

**Резюме**

**void** **HAL\_TIMER\_ChnETRResetConfig**( TIMER\_type\* TIMERx, TIMER\_CH\_type Channel, FunctionalState\_type NewState )

TIMER\_type\* TIMERx – указатель на структуру с регистрами таймера

TIMER\_CH\_type Channel – канала таймера (*TIMER\_CH1 – TIMER\_CH4*)

FunctionalState\_type NewState – состояние (*ENABLE, DISABLE*)

**Описание**

Функция управления сбросом от вывода входа внешней тактовой частоты ETR.

## HAL\_TIMER\_ChnBRKResetConfig

**Резюме**

**void** **HAL\_TIMER\_ChnBRKResetConfig**( TIMER\_type\* TIMERx, TIMER\_CH\_type Channel, FunctionalState\_type NewState )

TIMER\_type\* TIMERx – указатель на структуру с регистрами таймера

TIMER\_CH\_type Channel – канала таймера (*TIMER\_CH1 – TIMER\_CH4*)

FunctionalState\_type NewState – состояние (*ENABLE, DISABLE*)

**Описание**

Функция управления сбросом от вывода входа внешнего сигнала сброса BRK.

## HAL\_TIMER\_ChnREFFormatConfig

**Резюме**

**void** **HAL\_TIMER\_ChnREFFormatConfig**( TIMER\_type\* TIMERx, TIMER\_CH\_type Channel, TIMER\_PWM\_REF\_type Format )

TIMER\_type\* TIMERx – указатель на структуру с регистрами таймера

TIMER\_CH\_type Channel – канала таймера (*TIMER\_CH1 – TIMER\_CH4*)

TIMER\_PWM\_REF\_type Format – формат

**Описание**

Функция настройки формата REF сигнала канала у выбранного таймера. Формат Format описан в структуре TIMER\_ChnInit\_type в поле REF\_Format*.*

## HAL\_TIMER\_ChnCapturePrescalerConfig

**Резюме**

**void** **HAL\_TIMER\_ChnCapturePrescalerConfig**( TIMER\_type\* TIMERx, TIMER\_CH\_type Channel, TIMER\_Prescaler\_type Prescaler )

TIMER\_type\* TIMERx – указатель на структуру с регистрами таймера

TIMER\_CH\_type Channel – канала таймера (*TIMER\_CH1 – TIMER\_CH4*)

TIMER\_Prescaler\_type Prescaler – значение пред делителя частоты

**Описание**

Функция настройки пред делителя канала у выбранного таймера. Формат Prescaler описан в структуре TIMER\_ChnInit\_type в поле Prescaler*.*

## HAL\_TIMER\_ChnEventSourceConfig

**Резюме**

**void** **HAL\_TIMER\_ChnEventSourceConfig**( TIMER\_type\* TIMERx, TIMER\_CH\_type Channel, TIMER\_EventIn\_type EventSource )

TIMER\_type\* TIMERx – указатель на структуру с регистрами таймера

TIMER\_CH\_type Channel – канала таймера (*TIMER\_CH1 – TIMER\_CH4*)

TIMER\_EventIn\_type EventSource – вид события

**Описание**

Функция настройки входного события канала у выбранного таймера. Формат EventSource описан в структуре TIMER\_ChnInit\_type в поле EventSource*.*

## HAL\_TIMER\_ChnFilterConfig

**Резюме**

**void** **HAL\_TIMER\_ChnFilterConfig**( TIMER\_type\* TIMERx, TIMER\_CH\_type Channel, TIMER\_Filter\_type Filter )

TIMER\_type\* TIMERx – указатель на структуру с регистрами таймера

TIMER\_CH\_type Channel – канала таймера (*TIMER\_CH1 – TIMER\_CH4*)

TIMER\_Filter\_type Filter – параметры фильтра

**Описание**

Функция настройки входного фильтра канала у выбранного таймера. Формат Filter описан в структуре TIMER\_ChnInit\_type в поле FilterConf*.*

## HAL\_TIMER\_GetChnWriteComplete

**Резюме**

State\_type **HAL\_TIMER\_GetChnWriteComplete**( TIMER\_type\* TIMERx, TIMER\_CH\_type Channel )

TIMER\_type\* TIMERx – указатель на структуру с регистрами таймера

TIMER\_CH\_type Channel – канала таймера (*TIMER\_CH1 – TIMER\_CH4*)

**Описание**

Функция возвращает текущее состояние записи регистра CCR канала выбранного таймера:

*RESET* = 0

*SET* = 1

## HAL\_TIMER\_ChnCCR1\_EventSourceConfig

**Резюме**

**void** **HAL\_TIMER\_ChnCCR1\_EventSourceConfig**( TIMER\_type\* TIMERx, TIMER\_CH\_type Channel, TIMER\_EventIn\_type EventSource )

TIMER\_type\* TIMERx – указатель на структуру с регистрами таймера

TIMER\_CH\_type Channel – канала таймера (*TIMER\_CH1 – TIMER\_CH4*)

TIMER\_EventIn\_type EventSource – вид события

**Описание**

Функция настройки входного события CAP1 канала у выбранного таймера. Формат EventSource описан в структуре TIMER\_ChnInit\_type в поле EventSource*.*

## HAL\_TIMER\_ChnCCR1\_Cmd

**Резюме**

**void** **HAL\_TIMER\_ChnCCR1\_Cmd**( TIMER\_type\* TIMERx, TIMER\_CH\_type Channel, FunctionalState\_type NewState )

TIMER\_type\* TIMERx – указатель на структуру с регистрами таймера

TIMER\_CH\_type Channel – канала таймера (*TIMER\_CH1 – TIMER\_CH4*)

FunctionalState\_type NewState – состояние (*ENABLE, DISABLE*)

**Описание**

Функция управления работой регистра CCR1 канала у выбранного таймера.

## HAL\_TIMER\_ChnOutInit

**Резюме**

**void** **HAL\_TIMER\_ChnOutInit**( TIMER\_type\* TIMERx, **const** TIMER\_ChnOutInit\_type\* TIMER\_ChnOutInitStruct )

TIMER\_type\* TIMERx – указатель на структуру с регистрами таймера

**const** TIMER\_ChnOutInit\_type\* TIMER\_ChnOutInitStruct – указатель на структуру с параметрами выхода канала таймера

**Описание**

Функция настройки прямого и инверсного выхода канала выбранного таймера.

## HAL\_TIMER\_ChnOutStructInit

**Резюме**

**void** **HAL\_TIMER\_ChnOutStructInit**( TIMER\_ChnOutInit\_type\* TIMER\_ChnOutInitStruct )

TIMER\_ChnOutInit\_type\* TIMER\_ChnOutInitStruct – указатель на структуру с параметрами выхода канала таймера

**Описание**

Функция задает параметры инициализационной структуры выхода канала таймера значениями по умолчанию:

TIMER\_ChnOutInitStruct->DirOut\_Polarity = *TIMER\_CHOPolarity\_NonInverted*;

TIMER\_ChnOutInitStruct->DirOut\_Source = *TIMER\_CH\_OutSrc\_Only\_0*;

TIMER\_ChnOutInitStruct->DirOut\_Mode = *TIMER\_CH\_OutMode\_Input*;

TIMER\_ChnOutInitStruct->NegOut\_Polarity = *TIMER\_CHOPolarity\_NonInverted*;

TIMER\_ChnOutInitStruct->NegOut\_Source = *TIMER\_CH\_OutSrc\_Only\_0*;

TIMER\_ChnOutInitStruct->NegOut\_Mode = *TIMER\_CH\_OutMode\_Input*;

TIMER\_ChnOutInitStruct->DTG\_MainPrescaler = 0;

TIMER\_ChnOutInitStruct->DTG\_AuxPrescaler = 0;

TIMER\_ChnOutInitStruct->DTG\_ClockSource = *TIMER\_CH\_DTG\_ClkSrc\_TIMER\_CLK*;

## HAL\_TIMER\_ChnOutConfig

**Резюме**

**void** **HAL\_TIMER\_ChnOutConfig**( TIMER\_type\* TIMERx, TIMER\_CH\_type Channel, TIMER\_CHO\_SOURCE\_type OutSource, TIMER\_CHO\_MODE\_type Mode, TIMER\_CHO\_POLARITY\_type Polarity )

TIMER\_type\* TIMERx – указатель на структуру с регистрами таймера

TIMER\_CH\_type Channel – канала таймера (*TIMER\_CH1 – TIMER\_CH4*)

TIMER\_CHO\_SOURCE\_type OutSource – источник сигнала

TIMER\_CHO\_MODE\_type Mode – режим работы (вход/выход)

TIMER\_CHO\_POLARITY\_type Polarity – инверсия выхода

**Описание**

Функция настройки прямого выхода канала выбранного таймера. Значения аргументов описаны в структуре TIMER\_ChnOutInit\_type и соответствуют полям DirOut\_Source, DirOut\_Mode, DirOut\_Polarity.

## HAL\_TIMER\_ChnNOutConfig

**Резюме**

**void HAL\_TIMER\_ChnNOutConfig**( TIMER\_type\* TIMERx, TIMER\_CH\_type Channel, TIMER\_CHO\_SOURCE\_type OutSource, TIMER\_CHO\_MODE\_type Mode, TIMER\_CHO\_POLARITY\_type Polarity )

TIMER\_type\* TIMERx – указатель на структуру с регистрами таймера

TIMER\_CH\_type Channel – канала таймера (*TIMER\_CH1 – TIMER\_CH4*)

TIMER\_CHO\_SOURCE\_type OutSource – источник сигнала

TIMER\_CHO\_MODE\_type Mode – режим работы (вход/выход)

TIMER\_CHO\_POLARITY\_type Polarity – инверсия выхода

**Описание**

Функция настройки инвесного выхода канала выбранного таймера. Значения аргументов описаны в структуре TIMER\_ChnOutInit\_type и соответствуют полям NegOut\_Source, NegOut\_Mode, NegOut\_Polarity.

## HAL\_TIMER\_ChnOutDTGConfig

**Резюме**

**void HAL\_TIMER\_ChnOutDTGConfig**( TIMER\_type\* TIMERx, TIMER\_CH\_type Channel, uint16\_t MainPrescaler, uint16\_t AuxPrescaler, TIMER\_CH\_DTG\_CLKSRC\_type ClockSource )

TIMER\_type\* TIMERx – указатель на структуру с регистрами таймера

TIMER\_CH\_type Channel – канала таймера (*TIMER\_CH1 – TIMER\_CH4*)

uint16\_t MainPrescaler – основной делитель частоты

uint16\_t AuxPrescaler – предварительный делитель частоты

TIMER\_CH\_DTG\_CLKSRC\_type ClockSource – источник тактирования (*TIMER\_CH\_DTG\_ClkSrc\_TIMER\_CLK* или *TIMER\_CH\_DTG\_ClkSrc\_FDTS*)

**Описание**

Функция настройки генератора мертвого времени. Тактироваться генератор может как тактовой частотой таймера, так и частотой семплирования таймера.

DTG\_delay = MainPrescaler \* ( AuxPrescaler + 1 )

## HAL\_TIMER\_GetStatus

**Резюме**

uint32\_t **HAL\_TIMER\_GetStatus**( TIMER\_type\* TIMERx )

TIMER\_type\* TIMERx – указатель на структуру с регистрами таймера

**Описание**

Функция возвращает статусный регистр таймера.

## HAL\_TIMER\_GetFlagStatus

**Резюме**

State\_type **HAL\_TIMER\_GetFlagStatus**( TIMER\_type\* TIMERx, uint32\_t Flag )

TIMER\_type\* TIMERx – указатель на структуру с регистрами таймера

uint32\_t Flag ­– интересующие флаги таймера

**Описание**

Функция возвращает состояние интересующих флагов таймера. Флаги могут быть:

TIMER\_EVENT\_CNT\_ZERO – событие совпадения CNT с нулем

TIMER\_EVENT\_CNT\_ARR – событие совпадения CNT с ARR

TIMER\_EVENT\_ETR\_RISING\_EDGE – событие переднего фронта на входе ETR

TIMER\_EVENT\_ETR\_FALLING\_EDGE – событие заднего фронта на входе ETR

TIMER\_EVENT\_BRK – триггерированное по PCLK состояние входа BRK

TIMER\_EVENT\_CCR\_CAP\_CH1 – событие переднего фронта на входе CAP каналов таймера

TIMER\_EVENT\_CCR\_CAP\_CH2

TIMER\_EVENT\_CCR\_CAP\_CH3

TIMER\_EVENT\_CCR\_CAP\_CH4

TIMER\_EVENT\_CCR\_REF\_CH1 – событие переднего фронта на выходе REF каналов таймера

TIMER\_EVENT\_CCR\_REF\_CH2

TIMER\_EVENT\_CCR\_REF\_CH3

TIMER\_EVENT\_CCR\_REF\_CH4

TIMER\_EVENT\_CCR\_CAP1\_CH1 – событие переднего фронта на входе CAP1 каналов таймера

TIMER\_EVENT\_CCR\_CAP1\_CH2

TIMER\_EVENT\_CCR\_CAP1\_CH3

TIMER\_EVENT\_CCR\_CAP1\_CH4

Допускается передавать несколько событий через логическое ИЛИ:

**HAL\_TIMER\_GetFlagStatus**( LX\_TIMER0, TIMER\_EVENT\_CNT\_ZERO | TIMER\_EVENT\_CNT\_ARR | TIMER\_EVENT\_BRK )

Возвращаемое значение:

*RESET* = 0

*SET* = 1

## HAL\_TIMER\_ClearFlag

**Резюме**

**void** **HAL\_TIMER\_ClearFlag**( TIMER\_type\* TIMERx, uint32\_t Flags )

TIMER\_type\* TIMERx – указатель на структуру с регистрами таймера

uint32\_t Flags ­– флаги таймера

**Описание**

Функция производит сброс заданных флагов выбранного таймера. Параметр Flags подробно расписан в описании функции **HAL\_TIMER\_GetFlagStatus.**

## HAL\_TIMER\_DMACmd

**Резюме**

**void** **HAL\_TIMER\_DMACmd**( TIMER\_type\* TIMERx, uint32\_t TIMER\_DMASource, TIMER\_DMA\_CH\_type TIMER\_DMA\_Channel, FunctionalState\_type NewState )

TIMER\_type\* TIMERx – указатель на структуру с регистрами таймера

uint32\_t TIMER\_DMASource – источник запросов (флаг таймера)

TIMER\_DMA\_CH\_type TIMER\_DMA\_Channel – канал DMA

FunctionalState\_type NewState – состояние (*ENABLE, DISABLE*)

**Описание**

Функция настройки источника запросов к DMA от выбранного таймера. В качестве источника запросов указывается флаг таймера, подробно описанный в описании функции **HAL\_TIMER\_GetFlagStatus**.TIMER\_DMA\_Channel отвечает за выбор от какого канала таймера будет формироваться запрос:

*TIMER\_DMA\_CH1*

*TIMER\_DMA\_CH2*

*TIMER\_DMA\_CH3*

*TIMER\_DMA\_CH4*

*TIMER\_DMA\_ALL\_CH*

## HAL\_TIMER\_ITConfig

**Резюме**

**void** **HAL\_TIMER\_ITConfig**( TIMER\_type\* TIMERx, uint32\_t TIMER\_IT, FunctionalState\_type NewState )

TIMER\_type\* TIMERx – указатель на структуру с регистрами таймера

uint32\_t TIMER\_IT – источник запросов на прерывание (флаг таймера)

FunctionalState\_type NewState – состояние (*ENABLE, DISABLE*)

**Описание**

Функция настройки формирования прерываний от выбранного таймера. В качестве TIMER\_IT указывается флаг таймера, подробно описанный в описании функции **HAL\_TIMER\_GetFlagStatus.**

## HAL\_TIMER\_GetITStatus

**Резюме**

State\_type **HAL\_TIMER\_GetITStatus**( TIMER\_type\* TIMERx, uint32\_t TIMER\_IT )

TIMER\_type\* TIMERx – указатель на структуру с регистрами таймера

uint32\_t TIMER\_IT – интересующее прерывание (флаг таймера)

**Описание**

Функция возвращает статус интересующих прерываний выбранного таймера. В качестве TIMER\_IT указывается флаг таймера, подробно описанный в описании функции **HAL\_TIMER\_GetFlagStatus.**

Возвращаемое значение:

*RESET* = 0

*SET* = 1

# Пример работы

**#include** <hal\_1967VN044.h>

**#define** CoreClk\_kHz 100000

**#define** TIMER\_DMA\_CH 5

**#define** PWM\_DATA\_LEN 6

**#define** TIMER\_CCR1\_offset 0x4

uint32\_t ulPwm[ PWM\_DATA\_LEN ] = { 1, 3, 10, 30, 100, 300 };

**static** \_\_builtin\_quad pwm\_tcb;

**int** **main**( **void** )

{

uint64\_t ullData;

uint32\_t \*ptr = ( uint32\_t\* ) &pwm\_tcb;

TIMER\_CntInit\_type xTimerCnt;

TIMER\_ChnInit\_type xTimerCh;

TIMER\_ChnOutInit\_type xTimerChOut;

HAL\_PLL\_CoreSetup( CoreClk\_kHz );

// Разрешаем работу таймера 1

LX\_CMU->CFG1.b.GTMR1\_EN = 1;

// Настраиваем таймер 1

HAL\_TIMER\_CntStructInit( &xTimerCnt );

xTimerCnt.Period = 500;

xTimerCnt.Prescaler = 1000;

HAL\_TIMER\_CntInit( LX\_TIMER1, &xTimerCnt );

// Настраиваем 1 канал таймера 1

HAL\_TIMER\_ChnStructInit( &xTimerCh );

xTimerCh.ChNumber = *TIMER\_CH1*;

xTimerCh.REF\_Format = *TIMER\_CH\_REF\_Format6*;

xTimerCh.CCR\_UpdateMode = *TIMER\_Update\_On\_CNT\_Overflow*;

HAL\_TIMER\_ChnInit( LX\_TIMER1, &xTimerCh );

// Настраиваем выход 1 канала таймера 1

HAL\_TIMER\_ChnOutStructInit( &xTimerChOut );

xTimerChOut.ChNumber = *TIMER\_CH1*;

xTimerChOut.DirOut\_Mode = *TIMER\_CH\_OutMode\_Output*;

xTimerChOut.DirOut\_Source = *TIMER\_CH\_OutSrc\_REF*;

HAL\_TIMER\_ChnOutInit( LX\_TIMER1, &xTimerChOut );

// Настраиваем вывод используемый 1 прямым каналом таймера 1

HAL\_GPIO\_Init( LX\_GPIO\_PB, *GPIO\_PIN\_8*, *GPIO\_PinMode\_Alt* );

/\* ---------------------------- Настройка DMA -------------------------------------- \*/

HAL\_DMA\_Stop( TIMER\_DMA\_CH );

\*( ptr + 0 ) = ( uint32\_t ) ulPwm; // Источник информации

\*( ptr + 1 ) = ( PWM\_DATA\_LEN << 16 ) | 1; // Количество слов + инкремент адреса

\*( ptr + 2 ) = 0; // Количество попыток передачи по X + инкремент адреса X

\*( ptr + 3 ) = TCB\_NORMAL | // Длина передаваемых данных (операнда) в одном цикле обмена

TCB\_INTMEM; // Источник во внутренней памяти

HAL\_DMA\_CreateChannelDest( TIMER\_DMA\_CH, &pwm\_tcb, &pwm\_tcb );

HAL\_DMA\_RqstSet( TIMER\_DMA\_CH, *dmaTIMER1\_PWM1* );

// С помомщью регистра DCA указываем в какой регистр таймера загружать новые данные

ullData = DMA\_DCA\_INT\_MEM | ( base\_TIMER1 + TIMER\_CCR1\_offset );

HAL\_DMA\_WriteDCA( TIMER\_DMA\_CH, &ullData );

HAL\_DMA\_WriteDC( TIMER\_DMA\_CH, &pwm\_tcb );

// Разрешаем запросы к DMA от таймера 1

HAL\_TIMER\_ITConfig( LX\_TIMER1, TIMER\_EVENT\_CNT\_ARR, *ENABLE* );

HAL\_TIMER\_DMACmd( LX\_TIMER1, TIMER\_EVENT\_CNT\_ARR, *TIMER\_DMA\_CH1*, *ENABLE* );

HAL\_TIMER\_Cmd( LX\_TIMER1, *ENABLE* );

**while**( 1 )

{

HAL\_SYS\_WaitMs( 10 );

}

}

В данном примере показан вывод ШИМ с помощью DMA через 1 прямой канал таймера 1. Запрос к DMA возникает в момент совпадения CNT с ARR таймера.

Запросы от таймеров:

*dmaTIMERX\_PWM0* – запрос от всех 4 каналов

*dmaTIMERX\_PWM1* – запрос от 1 канала

*dmaTIMERX\_PWM2* – запрос от 2 канала

*dmaTIMERX\_PWM3* – запрос от 3 канала

*dmaTIMERX\_PWM4* – запрос от 4 канала