HAL DMA

# void HAL\_DMA\_WriteDC (int ch\_number, void \*qw\_tcb)

**Резюме**

**#include <hal\_1967VN028.h>**

**void** **HAL\_DMA\_WriteDC** (**int** ch\_number, **void** \*qw\_tcb)

**int** ch\_number – номер канала DMA (с 4 по 13)

**void** qw\_tcb – указатель на конфигурацию для регистра TCB, которая должна быть выровнена на границу квадрослова

**Описание**

Функция инициализирует выбранный канал DMA значениями, лежащими по указателю \*qw\_tcb (должен указывать на квадрослово, хранящее значения для инициализации DMA).

**Пример 1**

**#include <hal\_1967VN028.h>**

**#define** ADDRESS\_MEMORY 0x40000

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

**\_\_builtin\_quad** tcb;

**unsigned int** \*ptr;

ptr = (unsigned int \*)&tcb;

\*(ptr + 0) = ADDRESS\_MEMORY;

\*(ptr + 1) = (1024<<16) | 4;

\*(ptr + 2) = 0;

\*(ptr + 3) = TCB\_INTMEM | TCB\_QUAD;

**HAL\_DMA\_WriteDC** (4, &tcb);

**return** 0;

}

**Пример 2**

**#include <hal\_1967VN028.h>**

**#define** ADDRESS\_MEMORY 0x40000

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

**unsigned int \_\_attribute**((aligned(4 \* (32/\_\_CHAR\_BIT\_\_)))) tcb[4];

tcb[0] = ADDRESS\_MEMORY;

tcb[1] = (1024<<16) | 4;

tcb[2] = 0;

tcb[3] = TCB\_INTMEM | TCB\_QUAD;

**HAL\_DMA\_WriteDC** (5, &tcb);

**return** 0;

}

# void HAL\_DMA\_WriteDCS (int ch\_number, void \*qw\_tcb) void HAL\_DMA\_WriteDCD (int ch\_number, void \*qw\_tcb)

**Резюме**

**#include <hal\_1967VN028.h>**

**void** **HAL\_DMA\_WriteDCS**(**int** ch\_number, **void** \*qw\_tcb)

**void** **HAL\_DMA\_WriteDCD** (**int** ch\_number, **void** \*qw\_tcb)

**int** ch\_number – номер канала DMA общего назначения(с 0 по 3)

**void** qw\_tcb – указатель на конфигурацию для регистра TCB, которая должна быть выровнена на границу квадрослова

**Описание**

Функция инициализирует передатчик/приемник выбранного канал DMA общего назначения, значениями, лежащими по указателю \*qw\_tcb (должен указывать на квадрослово хранящее значения для инициализации DMA).

**Пример 1**

**#include <hal\_1967VN028.h>**

**#define** ADDRESS\_MEMORY\_TX 0x40000000

**#define** ADDRESS\_MEMORY\_RX 0x00040000

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

**\_\_builtin\_quad** tcb\_rx, tcb\_tx;

**unsigned int** \*ptr;

ptr = (unsigned int \*)&tcb\_rx;

\*(ptr + 0) = ADDRESS\_MEMORY\_RX;

\*(ptr + 1) = (1024<<16) | 4;

\*(ptr + 2) = 0;

\*(ptr + 3) = TCB\_INTMEM | TCB\_QUAD;

ptr = (unsigned int \*)&tcb\_tx;

\*(ptr + 0) = ADDRESS\_MEMORY\_TX;

\*(ptr + 1) = (1024<<16) | 4;

\*(ptr + 2) = 0;

\*(ptr + 3) = TCB\_EXTMEM | TCB\_QUAD;

**HAL\_DMA\_WriteDCD** (0, &tcb\_rx);

**HAL\_DMA\_WriteDCS** (0, &tcb\_tx);

**return** 0;

}

**Пример 2**

**#include <hal\_1967VN028.h>**

**#define** ADDRESS\_MEMORY\_TX 0x40000000

**#define** ADDRESS\_MEMORY\_RX 0x00040000

**unsigned int \_\_attribute**((aligned(4 \* (32/\_\_CHAR\_BIT\_\_)))) tcb\_rx[4], tcb\_tx[4];

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

tcb\_rx[0] = ADDRESS\_MEMORY\_RX;

tcb\_rx[1] = (1024<<16) | 4;

tcb\_rx[2] = 0;

tcb\_rx[3] = TCB\_INTMEM | TCB\_QUAD;

tcb\_tx[0] = ADDRESS\_MEMORY\_TX;

tcb\_tx[1] = (1024<<16) | 4;

tcb\_tx[2] = 0;

tcb\_tx[3] = TCB\_EXTMEM | TCB\_QUAD;

**HAL\_DMA\_WriteDCD** (3, &tcb\_rx);

**HAL\_DMA\_WriteDCS** (3, &tcb\_tx);

**return** 0;

}

# void HAL\_DMA\_ReadDC (int ch\_number, void \*qw\_tcb)

**Резюме**

**#include <hal\_1967VN028.h>**

**void** **HAL\_DMA\_ReadDC** (**int** ch\_number, **void** \*qw\_tcb)

**int** ch\_number – номер канала DMA (с 4 по 13)

\_\_builtin\_quad qw\_tcb – указатель на место сохранения регистра TCB

**Описание**

Функция сохраняет текущее состояния регистра TCB выбранного канала DMA по указателю (который должен указывать на квадрослово).

**Пример 1**

**#include <hal\_1967VN028.h>**

**#define** ADDRESS\_MEMORY 0x40000

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

**\_\_builtin\_quad** tcb;

**HAL\_DMA\_ReadDC** (9, &tcb);

**return** 0;

}

# void HAL\_DMA\_ReadDCS (int ch\_number, void \*qw\_tcb) void HAL\_DMA\_ReadDCD (int ch\_number, void \*qw\_tcb)

**Резюме**

**#include <hal\_1967VN028.h>**

**void** **HAL\_DMA\_ReadDCS** (**int** ch\_number, **void** \*qw\_tcb)

**void** **HAL\_DMA\_ReadDCD** (**int** ch\_number, **void** \*qw\_tcb)

**int** ch\_number – номер канала DMA общего назначения (с 0 по 3)

\_\_builtin\_quadqw\_tcb – указатель на место сохранения регистра TCB

**Описание**

Функция сохраняет текущее состояния регистра TCB выбранного канала DMA общего назначения по указателю (который должен указывать на квадрослово).

**Пример 1**

**#include <hal\_1967VN028.h>**

**#define** ADDRESS\_MEMORY 0x40000

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

**\_\_builtin\_quad** tcb;

**HAL\_DMA\_ReadDCS** (1, &tcb);

**HAL\_DMA\_ReadDCD** (2, &tcb);

**return** 0;

}

# void HAL\_DMA\_Stop(int ch\_number);

**Резюме**

**#include <hal\_1967VN028.h>**

**void** **HAL\_DMA\_Stop**(**int** ch\_number);

**int** ch\_number – номер канала DMA (с 0 по 13)

**Описание**

Функция останавливает (инициализирует нулями регистр TCB) выбранный канал DMA.

**Пример 1**

**#include <hal\_1967VN028.h>**

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

**HAL\_DMA\_Stop**(11);

**return** 0;

}

# int HAL\_DMA\_GetChannelStatus(int ch\_number)

**Резюме**

**#include <hal\_1967VN028.h>**

**void** **HAL\_DMA\_GetChannelStatus**(**int** ch\_number);

**int** ch\_number – номер канала DMA (с 0 по 13)

**Описание**

Функция возвращает статус заданного канала DMA.

* 0 – выключен;
* 1 – активен (работает);
* 2 – завершил работу;
* 3 – резерв;
* 4 – ошибка. Инициализация активного канала;
* 5 – ошибка. Запрещенная конфигурация канала;
* 6 – резерв;
* 7 – ошибка. Некорректный адрес.

**Пример 1**

**#include <hal\_1967VN028.h>**

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

**int** chStat;

chStat = **HAL\_DMA\_GetChannelStatus**(7);

**return** 0;

}

# int HAL\_DMA\_GetChannelStatusClear(int ch\_number)

**Резюме**

**#include <hal\_1967VN028.h>**

**void** **HAL\_DMA\_GetChannelStatus**(**int** ch\_number);

**int** ch\_number – номер канала DMA (с 0 по 13)

**Описание**

Функция возвращает статус заданного канала DMA, после чего очищает регистр статуса DMA.

* 0 – выключен;
* 1 – активен (работает);
* 2 – завершил работу;
* 3 – резерв;
* 4 – ошибка. Инициализация активного канала;
* 5 – ошибка. Запрещенная конфигурация канала;
* 6 – резерв;
* 7 – ошибка. Некорректный адрес.

**Пример 1**

**#include <hal\_1967VN028.h>**

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

**int** chStat;

chStat = **HAL\_DMA\_GetChannelStatusClear**(6);

**return** 0;

}

# int HAL\_DMA\_WaitForChannel(int ch\_number)

**Резюме**

**#include <hal\_1967VN028.h>**

**void** **HAL\_DMA\_WaitForChannel**(**int** ch\_number);

**int** ch\_number – номер канала DMA (с 0 по 13)

**Описание**

Функция исполняется пока идет прием/передача по выбранному каналу DMA. Возвращает следующие значения:

0 – канал завершил работу успешно

1 – обнаружена ошибка

2 – канал выключен

**Пример 1**

**#include <hal\_1967VN028.h>**

**#define** ADDRESS\_MEMORY 0x40000

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

**unsigned int \_\_attribute**((aligned(4 \* (32/\_\_CHAR\_BIT\_\_)))) tcb[4];

**int** errFlag;

tcb[0] = ADDRESS\_MEMORY;

tcb[1] = (1024<<16) | 4;

tcb[2] = 0;

tcb[3] = TCB\_INTMEM | TCB\_QUAD;

**HAL\_DMA\_WriteDC** (7, &tcb);

errFlag = **HAL\_DMA\_WaitForChannel** (7);

**return** 0;

}

# void HAL\_DMA\_GetDcCountX (int ch\_number)

**Резюме**

**#include <hal\_1967VN028.h>**

**int** **HAL\_DMA\_GetDcCountX** (**int** ch\_number);

**int** ch\_number – номер канала DMA (с 4 по 13)

**Описание**

Функция принимает номер канала DMA и возвращает, значение счетчика передаваемых/принимаемых данных этого канала

**Пример 1**

**#include <hal\_1967VN028.h>**

**#define** ADDRESS\_MEMORY 0x40000

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

{

**unsigned int** DC;

**unsigned int \_\_attribute**((aligned(4 \* (32/\_\_CHAR\_BIT\_\_)))) tcb[4];

tcb[0] = ADDRESS\_MEMORY;

tcb[1] = (1024<<16) | 4;

tcb[2] = 0;

tcb[3] = TCB\_INTMEM | TCB\_QUAD;

**HAL\_DMA\_WriteDC** (7, &tcb);

DC = **HAL\_DMA\_GetDcCountX** (6);

**return** 0;

}

# void HAL\_DMA\_CreateChannelDest (int ch\_number, void \*tcb\_current, void \*tcb\_next)

**Резюме**

**#include <hal\_1967VN028.h>**

**void** **HAL\_DMA\_CreateChannelDest** (**int** ch\_number, void \*tcb\_current, void \*tcb\_next);

**int** ch\_number – номер канала DMA. Задает следующий канал в цепочки DMA, для каналов с 4 по 11. Для каналов с 0 по 3, этот параметр не имеет значения.

**void** tcb\_current – указатель на текущую конфигурацию для регистров TCB, которая должна быть выровнена на границу квадрослова

**void** tcb\_next – указатель на следующую конфигурации для регистров TCB, которая должна быть выровнена на границу квадрослова

**Описание**

Функция принимает номер канала DMA, который будет следующим в цепочке. Указатель на текущую и следующую конфигурацию регистров TCB. Функция вызывается ДО записи первой конфигурации в цепочки в регистры TCB. Рассмотрим несколько примеров:

**Пример 1**

// Бесконечная передача по DMA одного и того же участка памяти в периферию по //каналу 4

**#include <hal\_1967VN028.h>**

**#define** ADDRESS\_MEMORY 0x40000

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

**unsigned int \_\_attribute**((aligned(4 \* (32/\_\_CHAR\_BIT\_\_)))) tcb[4];

tcb[0] = ADDRESS\_MEMORY;

tcb[1] = (1024<<16) | 4;

tcb[2] = 0;

tcb[3] = TCB\_INTMEM | TCB\_QUAD;

**HAL\_DMA\_CreateChannelDest**(4, &tcb, &tcb);

**HAL\_DMA\_WriteDC** (4, &tcb);

**return** 0;

}

**Пример 2**

// Два элемента в цепочка DMA зацикленные друг на друга (то есть сначала работает //tcb1 по 4 каналу, затем tcb2 по 5 каналу)

**#include <hal\_1967VN028.h>**

**#define** ADDRESS\_MEMORY 0x40000

**unsigned int \_\_attribute**((aligned(4 \* (32/\_\_CHAR\_BIT\_\_)))) tcb1[4], tcb2[4];

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

tcb1[0] = ADDRESS\_MEMORY;

tcb1[1] = (1024<<16) | 4;

tcb1[2] = 0;

tcb1[3] = TCB\_INTMEM | TCB\_QUAD;

tcb2[0] = ADDRESS\_MEMORY+0x40000;

tcb2[1] = (1024<<16) | 4;

tcb2[2] = 0;

tcb2[3] = TCB\_INTMEM | TCB\_QUAD;

**HAL\_DMA\_CreateChannelDest**(5, &tcb1, &tcb2);//после завершения передачи //данных по каналу с конфигурацией tcb1, начнётся передача данных по каналу 5, с //конфигурацией tcb2

**HAL\_DMA\_CreateChannelDest**(4, &tcb2, &tcb1); //после завершения передачи //данных по каналу с конфигурацией tcb2, начнётся передача данных по каналу 4, с //конфигурацией tcb1

**HAL\_DMA\_WriteDC** (4, &tcb1); //Включение DMA 4 (таким образом он //становится первым каналом в цепочке), после завершения работы канала 4, начнет //работать канал 5 с конфигурацией tcb2, затем после завершения работы канала 5, //начнет работать канал 4 и т.д.

**return** 0;

}

**Пример 3**

// Три элемента в цепочке, для каналов 0-3, для данной ситуации возьмем канал //0.

**#include** <hal\_1967VN028.h>

**#include** <stdint.h>

**#define** ADDRESS\_MEMORY 0x40000

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

**int** **\_\_attribute**((aligned(4 \* (32/\_\_CHAR\_BIT\_\_)))) tcbTx1[4], tcbTx2[4], tcbTx3[4], tcbRx[4];

tcbRx[0] = ADDRESS\_MEMORY;

tcbRx[1] = (1024<<16) | 0;

tcbRx[2] = (1024<<16);

tcbRx[3] = TCB\_INTMEM | TCB\_QUAD | TCB\_TWODIM | 0;

tcbTx1[0] = ADDRESS\_MEMORY+0x40000;

tcbTx1[1] = (1024<<16) | 0;

tcbTx1[2] = (1024<<16);

tcbTx1[3] = TCB\_INTMEM | TCB\_QUAD | TCB\_TWODIM | 0;

tcbTx2[0] = ADDRESS\_MEMORY+0x80000;

tcbTx2[1] = (1024<<16) | 0;

tcbTx2[2] = (1024<<16);

tcbTx2[3] = TCB\_INTMEM | TCB\_QUAD | TCB\_TWODIM | 0;

tcbTx3[0] = ADDRESS\_MEMORY+0xC0000;

tcbTx3[1] = (1024<<16) | 0;

tcbTx3[2] = (1024<<16);

tcbTx3[3] = TCB\_INTMEM | TCB\_QUAD | TCB\_TWODIM | 0;

HAL\_DMA\_CreateChannelDest(0, &tcbRx, &tcbRx); //Цепочка для приемника – //приемник бесконечно работает по конфигурации tcbRx

HAL\_DMA\_CreateChannelDest(1, &tcbTx1, &tcbTx2); //Цепочка tcbTx1 -> tcbTx2

HAL\_DMA\_CreateChannelDest(2, &tcbTx2, &tcbTx3); //Цепочка tcbTx2 -> tcbTx3

HAL\_DMA\_CreateChannelDest(3, &tcbTx3, &tcbTx1); //Цепочка tcbTx3 -> tcbTx1

//Обратите внимания, что при создании цепочки для каналов 0-3, входной //параметр ch\_number не имеет значения, т.к каналы 0-3 всегда переконфигурируют //новым элементом цепочки ТОЛЬКО самих себя. То есть канал, с которым будут работать //конфигурации, задается функциями HAL\_DMA\_WriteDCD и HAL\_DMA\_WriteDCS

HAL\_DMA\_WriteDCD(0,&tcbRx);// Включение приемника канал 0

HAL\_DMA\_WriteDCS(0,&tcbTx1);// Включения передатчика – канал 0, конфигурация //tcbTx1(т.е. tcbTx1 становится первым элементом в цепи)

}

# DMACopyStatus HAL\_DMA\_MemCopy32(int ch\_number, void \*a, void \*b, int size)

**Резюме**

**#include <hal\_1967VN028.h>**

**DMACopyStatus** **HAL\_DMA\_MemCopy32** (**int** ch\_number, void \*a, void \*b, **int** size);

**int** ch\_number – номер канала DMA общего назначения(с 0 по 3)

uint32\_t \*a – адрес передатчика

uint32\_t \*b – адрес приемник

**int** size – количество передаваемых слов

**Описание**

Функция передает массив по выбранному каналу DMA общего назначения. Передача производится словами, функции прерывания, цепочки, высокого приоритета, внешних запросов и двумерной передачи – отключены. Указывать можно как внешний, так и внутренней адрес передаваемого/принимаемого массива, исходя из этого функция автоматически определит режим передачи.

Функция возвращает код ошибки:

**Функция возвращает код ошибки:**

Таблица 1.3 Ошибки

|  |  |
| --- | --- |
| **Код ошибки** | **Описание** |
| 0 (dmaCopyValid) | Ошибок нет. Инициализация успешна. |
| 1 (dmaCopyAddrErr) | Некорректный адрес передаваемого/принимаемого массива |
| 2 (dmaCopyLengthErr) | Длинна передаваемых данных превышает максимальное значение |
| 3 (dmaCopyChNumErr) | Неверный канал DMA (разрешены канал с 0 по 3) |
| 4 (dmaCopyInitActCh) | Инициализация активного канала |
| 5 (dmaCopyCfgErr) | Запрещенная конфигурация канала |

**Ограничения**

size – не должно превышать значение 65535

**Пример**

//Передача по 2-му каналу DMA массива data\_tx32 по адресу массива data\_rx32, //находящихся во внутренней памяти

**#include <hal\_1967VN028.h>**

**#define** N 1024

**int** data\_tx32[N];

**int** data\_rx32[N];

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

**int** errFlag;

errFlag = HAL\_DMA\_MemCopy32 (2, &data\_tx32, &data\_rx32, N);

**return** 0;

}