HAL\_LINK

# int HAL\_LinkRx\_Enable (int link\_num, LinkRx\_Init\_type \*LRxInit)

**Резюме**

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

int **HAL\_LinkRx\_Enable**(**int** link\_num, LinkRx\_Init\_type \*LRxInit);

**int** link\_num – номер порта связи. Принимает значение 0 - 3.

LinkRx\_Init\_type \*LRxInit – указатель на структуры инициализации портов связи.

Табл.1

|  |  |  |
| --- | --- | --- |
| Имя поля | Описание | Значение |
| CheckSum | Настройка проверки CheckSum | * *Link\_CheckSum\_En*– разрешено * *Link\_CheckSum\_Dis*- запрещено |
| CheckBCMPI | Настройка анализа BCMPI | * *Link\_CheckBCMP\_En*– разрешено * *Link\_CheckBCMP\_Dis*- запрещено |
| DataSize | Размер шины для принимаемых данных | * *Link\_DataSize\_1bit* * *Link\_DataSize\_4bit* * *Link\_DataSize\_8bit* |
| TOIntEn | Разрешения прерывания по ситуации time out | * *Link\_TOIT\_En*– разрешено * *Link\_TOIT\_Dis*- запрещено |
| OVRIntEn | Разрешения прерывания при переполнении буфера приемника | * *Link\_OvrIT\_En*– разрешено * *Link\_OvrIT\_Dis*- запрещено |

**Описание**

Функция инициализирует приемник выбранного порта связи(link-порт). Для полей структуры необходимо использовать значения, в соответствии с табл.1. Функция возвращает -1 если номер порта связи указан неверно, в противном случае функция возвращает ноль.

**Пример 1**

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

**#define** LINK\_NUM 1

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

LinkRx\_Init\_type LRx;

int LInitSt;

LRx.CheckSum = Link\_CheckSum\_Dis;

LRx.CheckBCMPI = Link\_CheckBCMP\_Dis;

LRx.OVRIntEn = Link\_OvrIT\_Dis;

LRx.TOIntEn = Link\_TOIT\_Dis;

LRx.DataSize = Link\_DataSize\_4bit; //Link Enable

LInitSt = HAL\_LinkRx\_Enable(LINK\_NUM, &LRx);

}

# int HAL\_LinkTx\_Enable (int link\_num, LinkTx\_Init\_type \*LTxInit)

**Резюме**

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

int **HAL\_LinkTx\_Enable**(**int** link\_num, LinkTx\_Init\_type \*LTxInit);

**int** link\_num – номер порта связи. Принимает значение 0 - 3.

LinkTx\_Init\_type \*LTxInit – указатель на структуры инициализации портов связи.

Табл.2

|  |  |  |
| --- | --- | --- |
| Имя поля | Описание | Значение |
| CheckSum | Настройка проверки CheckSum | * *Link\_CheckSum\_En*– разрешено * *Link\_CheckSum\_Dis*- запрещено |
| CheckBCMPO | Разрешение формирования BCMPO | * *Link\_CheckBCMP\_En*– разрешено * *Link\_CheckBCMP\_Dis*- запрещено |
| DataSize | Размер шины для принимаемых данных | * *Link\_DataSize\_1bit* * *Link\_DataSize\_4bit* * *Link\_DataSize\_8bit* |
| TOIntEn | Разрешения прерывания по ситуации time out | * *Link\_TOIT\_En*– разрешено * *Link\_TOIT\_Dis*- запрещено |
| ClkDiv | Делитель клока LPLLCLK. Делит на 1,1.5,2, или 4, в зависимости от значения. Определяет частоту передачи LxCLKOUT. | * *Link\_ClkDiv\_1* * *Link\_ClkDiv\_1p5* * *Link\_ClkDiv\_2* * *Link\_ClkDiv\_4* |

**Описание**

Функция инициализирует передатчик выбранного порта связи(link-порт). Для полей структуры необходимо использовать значения, в соответствии с табл.2. Функция возвращает -1 если номер порта связи указан неверно, в противном случае функция возвращает ноль.

**Пример 1**

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

**#define** LINK\_NUM 1

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

LinkTx\_Init\_type LTx;

int LInitSt;

LTx.CheckBCMPO = Link\_CheckBCMP\_Dis;

LTx.CheckSum = Link\_CheckSum\_Dis;

LTx.TOIntEn = Link\_TOIT\_Dis;

LTx.ClkDiv = Link\_ClkDiv\_1;

LTx.DataSize = Link\_DataSize\_4bit; //Link Enable

LInitSt = HAL\_LinkTx\_Enable(LINK\_NUM, &LTx);

}

# int HAL\_LinkRx\_GetStat(int link\_num)

**Резюме**

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

**int HAL\_LinkRx\_GetStat**(**int** link\_num);

**int** link\_num – номер порта связи. Принимает значение 0 - 3.

**Описание**

Функция возвращает значение регистра статуса приемника заданного порта связи.

**Пример 1**

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

**#define** LINK\_NUM 0

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

**int** LRxST;

LRxST = HAL\_LinkRx\_GetStat(LINK\_NUM);

}

# int HAL\_LinkRx\_GetStatCl(int link\_num)

**Резюме**

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

**int HAL\_LinkRx\_GetStatCl**(**int** link\_num);

**int** link\_num – номер порта связи. Принимает значение 0 - 3.

**Описание**

Функция возвращает значение регистра статуса приемника заданного порта связи и сбрасывает в нем биты ошибок.

**Пример 1**

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

**#define** LINK\_NUM 0

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

**int** LRxST;

LRxST = HAL\_LinkRx\_GetStatCl(LINK\_NUM);

}

# int HAL\_LinkTx\_GetStat(int link\_num)

**Резюме**

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

**int HAL\_LinkTx\_GetStat**(**int** link\_num);

**int** link\_num – номер порта связи. Принимает значение 0 - 3.

**Описание**

Функция возвращает значение регистра статуса передатчика заданного порта связи.

**Пример 1**

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

**#define** LINK\_NUM 0

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

**int** LTxST;

LTxST = HAL\_LinkTx\_GetStat(LINK\_NUM);

}

# int HAL\_LinkTx\_GetStatCl(int link\_num)

**Резюме**

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

**int HAL\_LinkTx\_GetStatCl**(**int** link\_num);

**int** link\_num – номер порта связи. Принимает значение 0 - 3.

**Описание**

Функция возвращает значение регистра статуса передатчика заданного порта связи и сбрасывает в нем биты ошибок.

**Пример 1**

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

**#define** LINK\_NUM 0

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

**int** LTxST;

LTxST = HAL\_LinkTx\_GetStatCl(LINK\_NUM);

}

# void HAL\_LinkRx\_Disable(int link\_num)

**Резюме**

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

**void HAL\_LinkRx\_Disable**(**int** link\_num);

**int** link\_num – номер порта связи. Принимает значение 0 - 3.

**Описание**

Функция выключает приемник заданного порта связи.

**Пример 1**

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

**#define** LINK\_NUM 0

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

HAL\_LinkRx\_Disable(LINK\_NUM);

}

# void HAL\_LinkTx\_Disable(int link\_num)

**Резюме**

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

**void HAL\_LinkTx\_Disable**(**int** link\_num);

**int** link\_num – номер порта связи. Принимает значение 0 - 3.

**Описание**

Функция выключает передатчик заданного порта связи.

**Пример 1**

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

**#define** LINK\_NUM 0

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

HAL\_LinkTx\_Disable(LINK\_NUM);

}

# void HAL\_LinkDMA\_Receive(int link\_num, void \*tcb)

**Резюме**

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

**void HAL\_LinkDMA\_Receive**(**int** link\_num, **void** \*tcb);

**int** link\_num – номер порта связи. Принимает значение 0 - 3.

**void** \*tcb – указатель на конфигурацию для регистра TCB.

**Описание**

Функция запускает прием данных через заданный порт связи, по соответствующему каналу DMA. Например, для порта связи 0, для прима данных используется канал DMA 8. Также перезаписываются регистры TCB 8го канала DMA, на значения, заданные по указателю \*tcb. Для порта связи 1,2,3 используются каналы DMA 9,10,11. Для ожидания конца приема данных рекомендовано использовать функцию HAL\_DMA\_WaitForChannel(8+LINK\_NUM).

**Пример 1**

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

**#include** <stdint.h>

**#include** <stdio.h>

**#define** LINK\_NUM 0

**#define** DATA\_SIZE 1024

**unsigned** **int** data\_rx32[DATA\_SIZE] **\_\_attribute**((aligned(4)));

**void** **ERROR**()

{

printf("Error");

**while**(1) { };

};

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

LinkRx\_Init\_type LRx;

int LInitSt;

**unsigned** **int** tcbRx[4] **\_\_attribute**((aligned(4)));

**int** DMARxStat;

LRx.CheckSum = Link\_CheckSum\_Dis;

LRx.CheckBCMPI = Link\_CheckBCMP\_Dis;

LRx.OVRIntEn = Link\_OvrIT\_Dis;

LRx.TOIntEn = Link\_TOIT\_Dis;

LRx.DataSize = Link\_DataSize\_4bit; //Link Enable

LInitSt = HAL\_LinkRx\_Enable(LINK\_NUM, &LRx);

**if** (LInitSt != *0*) {ERROR();}

//TCB Init

tcbRx[0] = (uint32\_t) data\_rx32;

tcbRx[1] = (DATA\_SIZE<<16) | 4;

tcbRx[2] = 0;

tcbRx[3] = TCB\_INTMEM | TCB\_QUAD; //Работа с порта связи ВСЕГДА осуществляется //квадрословами.

//Receive Start

HAL\_LinkDMA\_Receive(LINK\_NUM, &tcbRx);

//DMA Wait

DMARxStat = HAL\_DMA\_WaitForChannel(8+LINK\_NUM); //TX

**if** (DMARxStat != 0) {ERROR();}

**return** (0);

}

# void HAL\_LinkDMA\_Transmit(int link\_num, void \*tcb)

**Резюме**

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

**void HAL\_LinkDMA\_Transmit**(**int** link\_num, **void** \*tcb);

**int** link\_num – номер порта связи. Принимает значение 0 - 3.

**void** \*tcb – указатель на конфигурацию для регистра TCB.

**Описание**

Функция запускает передачу данных через заданный порт связи, по соответствующему каналу DMA. Например, для порта связи 0, для передачи данных используется канал DMA 4. Также перезаписываются регистры TCB 4го канала DMA, на значения, заданные по указателю \*tcb. Для порта связи 1,2,3 используются каналы DMA 5,6,7. Для ожидания конца приема данных рекомендовано использовать функцию HAL\_DMA\_WaitForChannel(4+LINK\_NUM).

**Пример 1**

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

**#include** <stdint.h>

**#include** <stdio.h>

**#define** LINK\_NUM 1

**#define** DATA\_SIZE 1024

**unsigned** **int** data\_tx32[DATA\_SIZE] **\_\_attribute**((aligned(4)));

**void** **ERROR**()

{

printf("Error");

**while**(1) { };

};

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

LinkTx\_Init\_type LTx;

int LInitSt;

**unsigned** **int** tcbTx[4] **\_\_attribute**((aligned(4)));

**int** DMATxStat;

LTx.CheckBCMPO = Link\_CheckBCMP\_Dis;

LTx.CheckSum = Link\_CheckSum\_Dis;

LTx.TOIntEn = Link\_TOIT\_Dis;

LTx.ClkDiv = Link\_ClkDiv\_1;

LTx.DataSize = Link\_DataSize\_4bit; //Link Enable

LInitSt = HAL\_LinkTx\_Enable(LINK\_NUM, &LTx);

**if** (LInitSt != *0*) {ERROR();}

//TCB Init

tcbTx[0] = (uint32\_t) data\_tx32;

tcbTx[1] = (DATA\_SIZE<<16) | 4;

tcbTx[2] = 0;

tcbTx[3] = TCB\_INTMEM | TCB\_QUAD;//Работа с порта связи ВСЕГДА осуществляется //квадрословами.

//Transmit Start

HAL\_LinkDMA\_Transmit(LINK\_NUM, &tcbTx);

//DMA Wait

DMATxStat = HAL\_DMA\_WaitForChannel(4+LINK\_NUM); //TX

**if** (DMATxStat != 0) {ERROR();}

return(0);

}

# int HAL\_LinkCore\_Receive(int link\_num, void \*BuffPtr, int size,)

**Резюме**

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

**int HAL\_LinkDMA\_Receive**(**int** link\_num, **void** \* BuffPtr, **int** size);

**int** link\_num – номер порта связи. Принимает значение 0 - 3.

**void** \*BuffPtr – указатель на место сохранения принимаемых данных

**int** size – размер принимаемых данных(в словах, т.е должен быть кратен 4)

**Описание**

Функция осуществляет прием данных посредством чтения буфера приемника порта связи и сохраняет их по указателю BuffPtr. В случае установления бита ошибки в регистре статуса приемника, функция прерывается и возвращает значение -1. В случае успешного приема данных, функция возвращает 0.

**Пример 1**

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

**#include** <stdio.h>

**#include** <stdint.h>

**#define** LINK\_NUM 2

**#define** DATA\_SIZE 1024

**int** data\_rx32[DATA\_SIZE];

**void** **ERROR**()

{

**while**(1) { };

};

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

LinkRx\_Init\_type LRx;

**int** LInitSt;

**int** RxStat;

**uint32\_t** \*ptrDataRx;

LRx.CheckSum = *Link\_CheckSum\_Dis*;

LRx.CheckBCMPI = *Link\_CheckBCMP\_Dis*;

LRx.OVRIntEn = *Link\_OvrIT\_Dis*;

LRx.TOIntEn = *Link\_TOIT\_Dis*;

LRx.DataSize = *Link\_DataSize\_4bit*;

//Link Enable

LInitSt = HAL\_LinkRx\_Enable(LINK\_NUM, &LRx);

**if** (LInitSt != 0) {ERROR();}

//Receive Data

ptrDataRx = (uint32\_t \*) data\_rx32;

**for**(**int** i=0;i<DATA\_SIZE/4;i++){

RxStat = HAL\_LinkCore\_Receive(LINK\_NUM, (**void** \*)ptrDataRx, 4);

ptrDataRx+=4\*(sizeof(uint32\_t));

**if** (RxStat != 0) {ERROR();}

}

}

# int HAL\_LinkCore\_Transmit(int link\_num, void \*BuffPtr, int size,)

**Резюме**

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

**int HAL\_LinkDMA\_Transmit**(**int** link\_num, **void** \* BuffPtr, **int** size );

**int** link\_num – номер порта связи. Принимает значение 0 - 3.

**void** \*BuffPtr – указатель на начало передаваемых данных

**int** size – размер передаваемых данных(в словах, т.е должен быть кратен 4)

**Описание**

Функция осуществляет передачу данных из BuffPtr, посредством записи в буфер передатчика порта связи. В случае установления бита ошибки в регистре статуса передатчика, функция прерывается и возвращает значение -1. В случае успешной передачи данных, функция возвращает 0.

**Пример 1**

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

**#include** <stdio.h>

**#include** <stdint.h>

**#define** LINK\_NUM 2

**#define** DATA\_SIZE 1024

**int** data\_tx32[DATA\_SIZE];

**void** **ERROR**()

{

**while**(1) { };

};

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

LinkTx\_Init\_type LTx;

**int** LInitSt;

**unsigned int** \* ptrDataTx;

**int** TxStat;

LTx.CheckBCMPO = *Link\_CheckBCMP\_Dis*;

LTx.CheckSum = *Link\_CheckSum\_Dis*;

LTx.TOIntEn = *Link\_TOIT\_Dis*;

LTx.ClkDiv = *Link\_ClkDiv\_1*;

LTx.DataSize = *Link\_DataSize\_4bit*;

//Link Enable

LInitSt = HAL\_LinkTx\_Enable(LINK\_NUM, &LTx);

**if** (LInitSt != 0) {ERROR();}

//Transmit Data

ptrDataTx = (uint32\_t \*) data\_tx32;

**for**(**int** i=0;i<DATA\_SIZE/4;i++){

TxStat = HAL\_LinkCore\_Transmit(LINK\_NUM, (**void** \*)ptrDataTx, 4);

ptrDataTx+=4;

**if** (TxStat != 0) {ERROR();}

}

}