# Описание

Данная документация описывает Hardware Abstraction Layer для процессора 1967ВН044. Документация представляет из себя набор файлов, где каждый документ описывает один конкретный модуль HAL.

# Начало работы с HAL

Для работы HAL необходимо, чтобы в проекте был конфигурационный файл «1967VN044\_HAL\_config.h». Пользователь должен самостоятельно его создать и определить в нём внешнюю частоту – макрос XTI\_KHZ. В поставке HAL есть пример такого конфигурационного файла - «1967VN044\_HAL\_config\_sample.h». Не подключайте заголовочный файл примера в своих проектах!

Возможно, среда разработки CM\_LYNX потребует добавить "${workspace\_loc:\${ProjName}}" в Includes проекта.

# Описание используемых функций

## HAL\_AUDIO\_I2SInit

**Резюме**

**void** **HAL\_AUDIO\_I2SInit** ( Audio\_type\* **const** AudioX, Audio\_i2s\_type\* I2sConf )

Audio\_type\* **const** AudioX – указатель на структуру с регистрами аудиокодека

Audio\_i2s\_type\* I2sConf – указатель на структуру с конфигурационными настройками

**Описание**

Функция инициализации контроллера аудиокодека параметрами из структуры I2sConf. Функция настраивает требуемые регистры аудиокодека и используемые кодеком пины порта ввода-вывода.

## HAL\_AUDIO\_StartDMA\_TX

**Резюме**

**void** **HAL\_AUDIO\_StartDMA\_TX**( uint32\_t channel, Audio\_type\* **const** AudioX, **void** \*pvDataBuf, uint16\_t usSize, **void**\* pvDmaIsrHandler )

uint32\_t channel – номер канала DMA (от 4 до 7)

Audio\_type\* **const** AudioX – указатель на структуру с регистрами аудиокодека

**void** \*pvDataBuf – указатель на буфер с данными

uint16\_t usSize – размер буфера для передачи

**void**\* pvDmaIsrHandler – указатель на функцию обработчика прерывания, если не используется передаем 0.

**Описание**

Функция используется для инициализации передачи аудиоданных с помощью DMA.

## HAL\_AUDIO\_StartDMA\_RX

**Резюме**

**void** **HAL\_AUDIO\_StartDMA\_RX**( uint32\_t channel, Audio\_type\* **const** AudioX, **void** \*pvDataBuf, uint16\_t usSize, **void**\* pvDmaIsrHandler )

uint32\_t channel – номер канала DMA (от 8 до 11)

Audio\_type\* **const** AudioX – указатель на структуру с регистрами аудиокодека

**void** \*pvDataBuf – указатель на буфер с данными

uint16\_t usSize – размер буфера для приёма

**void**\* pvDmaIsrHandler – указатель на функцию обработчика прерывания, если не используется передаем 0.

**Описание**

Функция используется для инициализации приёма аудиоданных с помощью DMA.

## HAL\_AUDIO\_StopDMA

**Резюме**

**void** **HAL\_AUDIO\_StopDMA**( Audio\_type\* **const** AudioX )

Audio\_type\* **const** AudioX – указатель на структуру с регистрами аудиокодека

**Описание**

Функция используется для прекращения передачи аудиоданных с помощью DMA.

## HAL\_AUDIO\_StartDMA\_TX\_2BUF

**Резюме**

**void** **HAL\_AUDIO\_StartDMA\_TX\_2BUF**( uint32\_t channel, Audio\_type\* **const** AudioX, **void** \*pvDataBufA, **void** \*pvDataBufB, uint16\_t usSize, **void**\* pvDmaIsrHandler )

uint32\_t channel – номер канала DMA (от 4 до 7)

Audio\_type\* **const** AudioX – указатель на структуру с регистрами аудиокодека

**void** \*pvDataBufA – указатель на 1-й буфер с данными

**void** \*pvDataBufB – указатель на 2-й буфер с данными

uint16\_t usSize – размер буфера для передачи

**void**\* pvDmaIsrHandler – указатель на функцию обработчика прерывания, если не используется передаем 0.

**Описание**

Функция используется для инициализации передачи аудиоданных с помощью DMA. Используется цепочка с двумя буферами. Функция предназначена для обеспечения возможности обработки аудиоданных – пока данные в одном буфере декодируются, уже готовые данные из другого буфера передаются.

## HAL\_AUDIO\_StartDMA\_RX\_2BUF

**Резюме**

**void** **HAL\_AUDIO\_StartDMA\_RX\_2BUF**( uint32\_t channel, Audio\_type\* **const** AudioX, **void** \*pvDataBufA, **void** \*pvDataBufB, uint16\_t usSize, **void**\* pvDmaIsrHandler )

uint32\_t channel – номер канала DMA (от 8 до 11)

Audio\_type\* **const** AudioX – указатель на структуру с регистрами аудиокодека

**void** \*pvDataBufA – указатель на 1-й буфер с данными

**void** \*pvDataBufB – указатель на 2-й буфер с данными

uint16\_t usSize – размер буфера для приёма

**void**\* pvDmaIsrHandler – указатель на функцию обработчика прерывания, если не используется передаем 0.

**Описание**

Функция используется для инициализации приёма аудиоданных с помощью DMA. Используется цепочка с двумя буферами. Функция предназначена для обеспечения возможности обработки аудиоданных – пока данные в одном буфере принимаются, данные в другом буфере кодируются.

# Пример работы

**#include** "hal\_1967VN044.h"

**#define** CORECLK\_KHZ 200000

uint32\_t SomeBuff[ BUFF\_SIZE ]; // Буфер с аудиоданными

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

{

Audio\_i2s\_type xI2Sconf;

HAL\_PLL\_CoreSetup( CORECLK\_KHZ );

xI2Sconf.Role = ROLE\_SLAVE;

xI2Sconf.Mode = MODE\_I2S;

xI2Sconf.Standart = STANDART\_PHILLIPS;

xI2Sconf.TfsPos = 0;

xI2Sconf.SwapLR = 0;

xI2Sconf.DataLen = 16;

xI2Sconf.AudioFreq = SIN\_AUDIO\_FREQ;

LX\_CMU->CFG8.b.I2S0\_DIS = 0; // Подаем тактирование на I2S0

HAL\_AUDIO\_StartDMA\_TX( DMA\_AUDIO\_CHANNEL, LX\_AUDIO0, SomeBuff, BUFF\_SIZE, 0 );

HAL\_AUDIO\_I2SInit( LX\_AUDIO0, &xI2Sconf );

// Теперь I2S готов принимать запросы от внешнего кодека

}

В данном примере происходит инициализация аудиокодека в роли подчиненного и режиме i2s. Аудиоданные для выдачи должны находится в буфере SomeBuff. Дополнительно нужно проинициализировать используемую микросхему i2s ЦАП, чтобы она запрашивала данные из нашего аудиокодека. После выполнения функции HAL\_AUDIO\_I2SInit, аудиокодек готов выдавать данные по запросу мастера. Как только будет выдан весь буфер, будет сгенерировано прерывание (если был задан обработчик прерывания) и начнется повторная выдача буфера с аудиоданными.