



RTOS DMAC 开发指南

版本号: 1.0
发布日期: 2020.7.9

版本历史

版本号	日期	制/修订人	内容描述
1.0	2020.7.9	AWA1636	1. 初版

ALLWINER®

目 录

1 前言	1
1.1 文档简介	1
1.2 目标读者	1
1.3 适用范围	1
2 模块介绍	2
2.1 模块功能介绍	2
2.2 相关术语介绍	2
2.2.1 硬件术语	2
2.2.2 软件术语	2
2.3 模块配置介绍	2
2.3.1 platform 配置说明	2
2.3.2 kernel menuconfig 配置说明	3
2.4 源码结构介绍	3
2.5 驱动框架介绍	4
3 模块接口说明	5
3.1 hal_dma_chan_status_t hal_dma_chan_request	5
3.2 hal_dma_status_t hal_dma_chan_free	6
3.3 hal_dma_status_t hal_dma_chan_desc_free	6
3.4 hal_dma_status_t hal_dma_prep_cyclic	6
3.5 hal_dma_status_t hal_dma_prep_memcpy	7
3.6 hal_dma_status_t hal_dma_prep_device	7
3.7 hal_dma_status_t hal_dma_callback_install	7
3.8 hal_dma_status_t hal_dma_slave_config	8
3.9 enum dma_status hal_dma_tx_status	8
3.10 hal_dma_status_t hal_dma_start	9
3.11 hal_dma_status_t hal_dma_stop	9
3.12 void hal_dma_init(void)	9
3.13 void *dma_alloc_coherent	9
3.14 void dma_free_coherent	10
4 模块使用范例	11

1 前言

1.1 文档简介

介绍 RTOS 中 DMA 驱动接口及使用方法，为 DMA 的使用者提供参考。

1.2 目标读者

DMA 驱动、及应用层的开发/维护人员。

1.3 适用范围

表 1-1: 适用产品列表

产品名称	内核版本	驱动文件
F133	Melis	hal_dma.c
V833	Melis	hal_dma.c
R328	FreeRTOS	hal_dma.c
R329-DSP	FreeRTOS	hal_dma.c

2 模块介绍

2.1 模块功能介绍

BSP DMA 驱动主要实现设备驱动的底层细节，并为上层提供一套标准的 API 接口以供使用。

2.2 相关术语介绍

2.2.1 硬件术语

术语	解释说明
DMA	Direct Memory Access, 直接存储器存取

2.2.2 软件术语

术语	解释说明
HAL	Hardware Abstraction Layer, 硬件抽象层
RTOS	Real Time Operating System, 实时操作系统
GPIO	General Purpose Input/Output, 通用输入输出

2.3 模块配置介绍

2.3.1 platform 配置说明

在不同的 Sunxi 硬件平台中，DMA 设计不同，但平台配置文件的信息基本类似，如下：

```
#define SUNXI_DMACH_PBASE    0x03002000
#define DMA_IRQ_NUM          42

/*
 * The source DRQ type and port corresponding relation
```

```

*/
#define DRQSRC_SRAM          0
#define DRQSRC_SDRAM        0
#define DRQSRC_DAUDIO_0_RX  3
#define DRQSRC_DAUDIO_1_RX  4
#define DRQSRC_AUDIO_CODEC  6
...
/*
 * The destination DRQ type and port corresponding relation
 */
#define DRQDST_SRAM          0
#define DRQDST_SDRAM        0
#define DRQDST_DAUDIO_0_TX  3
#define DRQDST_DAUDIO_1_TX  4
#define DRQDST_AUDIO_CODEC  6
...

```

其中：

1. SUNXI_DMACH_PBASE，表示 DMACH 基地址；
2. DMA_IRQ_NUM，表示 DMACH 中断号；
3. DRQSRC_XXX，表示源 DRQ 号；
4. DRQDST_XXX，表示目的 DRQ 号；

2.3.2 kernel menuconfig 配置说明

Melis 版本：make menuconfig 配置路径：

Kernel Setup → Drivers Setup → SoC HAL Drivers → DMA Devices →

FreeRTOS 版本：mrtos_menuconfig 配置路径：

Drivers Options → soc related device drivers → DMA Devices →

```

[*] enable dma driver
[*] enable dma hal APIs test command

```

图 2-1: DMA menuconfig

2.4 源码结构介绍

```

hal/source/dma/ ---- 驱动源码
├─ hal_dma.c
├─ Kconfig
├─ Makefile
├─ platform
└─ dma-sun8iw18.h

```

```

├── dma-sun8iw19.h
├── dma-sun8iw20.h
└── platform-dma.h

include/hal/ ---- 驱动APIs声明头文件
├── hal_dma.h

hal/test/dma/ ---- 驱动APIs测试代码
├── Makefile
└── test_dma.c
    
```

2.5 驱动框架介绍

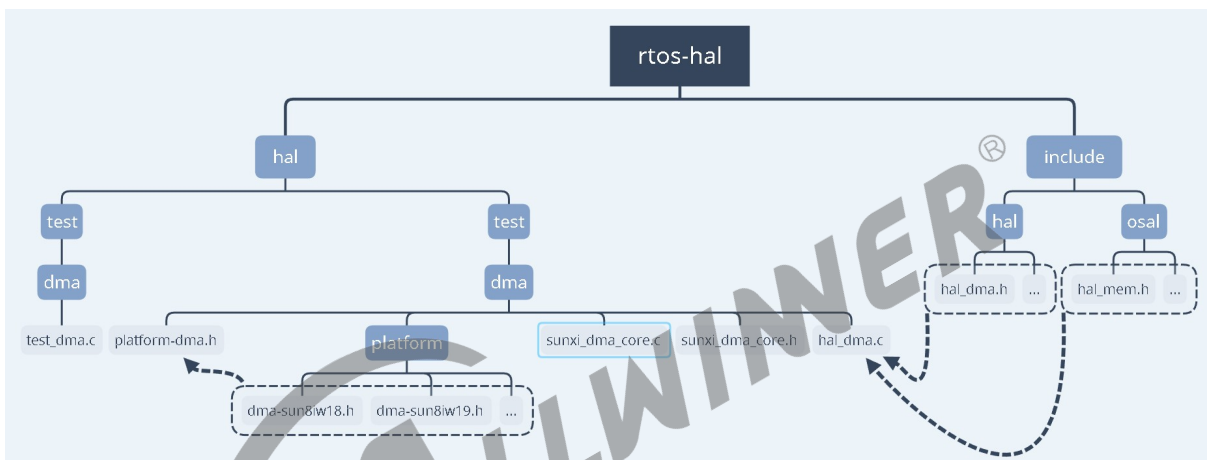


图 2-2: DMA 驱动框架

3 模块接口说明

需包含头文件：

```
#include <hal_dma.h>
```

API	解释说明
hal_dma_chan_request	申请 DMA 通道
hal_dma_chan_free	释放 DMA 通道
hal_dma_chan_desc_free	释放 DMA 通道描述符
hal_dma_prep_cyclic	初始化环形 DMA 传输
hal_dma_prep_memcpy	初始化 memory to memory DMA 传输
hal_dma_prep_device	初始化通用 DMA 传输
hal_dma_callback_install	注册 DMA 回调函数
hal_dma_slave_config	配置 DMA 描述符传输信息
hal_dma_tx_status	获取 DMA 发送状态
hal_dma_start	启动 DMA 传输
hal_dma_stop	停止 DMA 传输
hal_dma_init	初始化 DMA 控制器驱动
dma_alloc_coherent	申请一致性内存
dma_free_coherent	释放一致性内存

3.1 hal_dma_chan_status_t hal_dma_chan_request

- 原型：hal_dma_chan_status_t hal_dma_chan_request(struct sunxi_dma_chan **dma_chan)
- 作用：申请 DMA 通道
- 参数：
 - dma_chan: 存放 DMA 通道的指针变量
- 返回：
 - HAL_DMA_CHAN_STATUS_BUSY: 申请失败
 - HAL_DMA_CHAN_STATUS_FREE: 申请成功

3.2 hal_dma_status_t hal_dma_chan_free

- 原型：hal_dma_status_t hal_dma_chan_free(struct sunxi_dma_chan *chan)
- 作用：释放 DMA 通道
- 参数：
 - chan: 要释放的 DMA 通道结构体指针变量
- 返回：
 - HAL_DMA_STATUS_ERROR: 失败
 - HAL_DMA_STATUS_OK: 成功

3.3 hal_dma_status_t hal_dma_chan_desc_free

- 原型：hal_dma_status_t hal_dma_chan_desc_free(struct sunxi_dma_chan *chan)
- 作用：释放 DMA 通道描述符
- 参数：
 - chan: 要释放的 DMA 通道结构体指针变量
- 返回：
 - HAL_DMA_STATUS_ERROR: 失败
 - HAL_DMA_STATUS_OK: 成功

3.4 hal_dma_status_t hal_dma_prep_cyclic

- 原型：hal_dma_status_t hal_dma_prep_cyclic(struct sunxi_dma_chan *chan, uint32_t buf_addr, uint32_t buf_len, uint32_t period_len, enum dma_transfer_direction dir)
- 作用：初始化环形 DMA 传输
- 参数：
 - chan: DMA 通道结构体指针变量
 - buf_addr: 数据缓冲区
 - buf_len: 数据缓冲区长度
 - period_len: 单次 DMA 搬运长度
 - dir: DMA 传输方向
- 返回：
 - HAL_DMA_STATUS_INVALID_PARAMETER: 参数非法
 - HAL_DMA_STATUS_ERROR: 失败
 - HAL_DMA_STATUS_OK: 成功

3.5 hal_dma_status_t hal_dma_prep_memcpy

- 原型: hal_dma_status_t hal_dma_prep_memcpy(struct sunxi_dma_chan *chan, uint32_t dest, uint32_t src, uint32_t len)
- 作用: 初始化 memory to memory DMA 传输
- 参数:
 - chan: DMA 通道结构体指针变量
 - dest: 目的地址
 - src: 源地址
 - len: 传输长度
- 返回:
 - HAL_DMA_STATUS_INVALID_PARAMETER: 参数非法
 - HAL_DMA_STATUS_ERROR: 失败
 - HAL_DMA_STATUS_OK: 成功

3.6 hal_dma_status_t hal_dma_prep_device

- 原型: hal_dma_status_t hal_dma_prep_device(struct sunxi_dma_chan *chan, uint32_t dest, uint32_t src, uint32_t len, enum dma_transfer_direction dir)
- 作用: 初始化通用 DMA 传输
- 参数:
 - chan: DMA 通道结构体指针变量
 - dest: 目的地址
 - src: 源地址
 - len: 传输长度
 - dir: DMA 传输方向
- 返回:
 - HAL_DMA_STATUS_INVALID_PARAMETER: 参数非法
 - HAL_DMA_STATUS_ERROR: 失败
 - HAL_DMA_STATUS_OK: 成功

3.7 hal_dma_status_t hal_dma_callback_install

- 原型: hal_dma_status_t hal_dma_callback_install(struct sunxi_dma_chan *chan, dma_callback callback, void *callback_param)
- 作用: 注册 DMA 回调函数

- 参数：
 - chan:DMA 通道结构体指针变量
 - callback: 回调函数 handler
 - callback_param: 回调函数传参
- 返回：
 - HAL_DMA_STATUS_INVALID_PARAMETER: 参数非法
 - HAL_DMA_STATUS_OK: 成功

3.8 hal_dma_status_t hal_dma_slave_config

- 原型: hal_dma_status_t hal_dma_slave_config(struct sunxi_dma_chan *chan, struct dma_slave_config *config)
- 作用: 配置 DMA 描述符传输信息
- 参数：
 - chan:DMA 通道结构体指针变量
 - config:DMA 描述符结构体指针变量
- 返回：
 - HAL_DMA_STATUS_INVALID_PARAMETER: 参数非法
 - HAL_DMA_STATUS_OK: 成功

3.9 enum dma_status hal_dma_tx_status

- 原型: enum dma_status hal_dma_tx_status(struct sunxi_dma_chan *chan, uint32_t *left_size)
- 作用: 获取 DMA 发送状态
- 参数：
 - chan:DMA 通道结构体指针变量
 - left_size: 存放剩余长度的指针变量
- 返回：
 - DMA_INVALID_PARAMETER: 参数非法
 - DMA_IN_PROGRESS: 正在进行
 - DMA_COMPLETE: 传输完成

3.10 hal_dma_status_t hal_dma_start

- 原型：hal_dma_status_t hal_dma_start(struct sunxi_dma_chan *chan)
- 作用：启动 DMA 传输
- 参数：
 - chan:DMA 通道结构体指针变量
- 返回：
 - HAL_DMA_STATUS_INVALID_PARAMETER: 参数非法
 - HAL_DMA_STATUS_ERROR: 失败
 - HAL_DMA_STATUS_OK: 成功

3.11 hal_dma_status_t hal_dma_stop

- 原型：hal_dma_status_t hal_dma_stop(struct sunxi_dma_chan *chan)
- 作用：停止 DMA 传输
- 参数：
 - chan:DMA 通道结构体指针变量
- 返回：
 - HAL_DMA_STATUS_INVALID_PARAMETER: 参数非法
 - HAL_DMA_STATUS_ERROR: 失败
 - HAL_DMA_STATUS_OK: 成功

3.12 void hal_dma_init(void)

- 原型：void hal_dma_init(void)
- 作用：初始化 DMA 控制器驱动
- 参数：
 - void
- 返回：
 - void

3.13 void *dma_alloc_coherent

- 原型：void *dma_alloc_coherent(size_t size)

- 作用：申请一致性内存
- 参数：
 - 参数 1: 申请内存的大小
- 返回：
 - ptr: 内存缓冲区指针

3.14 void dma_free_coherent

- 原型：void dma_free_coherent(void *addr)
- 作用：释放一致性内存
- 参数：
 - addr: 内存缓冲区指针
- 返回：
 - void



4 模块使用范例

可参考驱动 APIs 测试代码 (hal/test/dma/)。






著作权声明

版权所有 © 2021 珠海全志科技股份有限公司。保留一切权利。

本档及内容受著作权法保护，其著作权由珠海全志科技股份有限公司（“全志”）拥有并保留一切权利。

本档是全志的原创作品和版权财产，未经全志书面许可，任何单位和个人不得擅自摘抄、复制、修改、发表或传播本档内容的部分或全部，且不得以任何形式传播。

商标声明

、 **全志科技** （不完全列举）均为珠海全志科技股份有限公司的商标或者注册商标。在本档描述的产品中出现的其它商标，产品名称，和服务名称，均由其各自所有人拥有。

免责声明

您购买的产品、服务或特性应受您与珠海全志科技股份有限公司（“全志”）之间签署的商业合同和条款的约束。本档中描述的全部或部分产品、服务或特性可能不在您所购买或使用的范围内。使用前请认真阅读合同条款和相关说明，并严格遵循本档的使用说明。您将自行承担任何不当使用行为（包括但不限于如超压，超频，超温使用）造成的不利后果，全志概不负责。

本档作为使用指导仅供参考。由于产品版本升级或其他原因，本档内容有可能修改，如有变更，恕不另行通知。全志尽全力在本档中提供准确的信息，但并不确保内容完全没有错误，因使用本档而发生损害（包括但不限于间接的、偶然的、特殊的损失）或发生侵犯第三方权利事件，全志概不负责。本档中的所有陈述、信息和建议并不构成任何明示或暗示的保证或承诺。

本档未以明示或暗示或其他方式授予全志的任何专利或知识产权。在您实施方案或使用产品的过程中，可能需要获得第三方的权利许可。请您自行向第三方权利人获取相关的许可。全志不承担也不代为支付任何关于获取第三方许可的许可费或版税（专利税）。全志不对您所使用的第三方许可技术做出任何保证、赔偿或承担其他义务。