HAL V2 CCU
开发指南

版本号: 1.0

发布日期: 2025.07.15

版本历史

版本号	日期	制/修订人	内容描述
1.0	2025.07.15	AWA1982	初始版本

版权所有 © 珠海全志科技股份有限公司。保留一切权利

目　录

版权所有 © 珠海全志科技股份有限公司。保留一切权利

版权所有 © 珠海全志科技股份有限公司。保留一切权利

表　格

版权所有 © 珠海全志科技股份有限公司。保留一切权利

1 前言

1.1 文档简介

本文档介绍 HAL V2 时钟驱动的接口及使用方法。

1.2 目标读者

- CCU 驱动开发工程师
- CCU 驱动使用工程师

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
T153	FreeRTOS/Baremetal	hal_n_clk.c
T536	FreeRTOS/Baremetal	hal_n_clk.c

1.4 文档约定

1 标志说明

⚠ 注意

提醒操作中应注意的事项。不当的操作可能会损坏器件，影响可靠性、降低性能等。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

说明

助准确理解文中指令、正确实施操作而提供的补充或强调信息。

技巧

一些容易忽视的小功能、技巧。了解这些功能或技巧能帮助解决特定问题或者节省操作时间。

1.4.2 地址与数据描述方法约定

本文档在描述地址、数据时遵循如下约定：

表 1-2: 地址与数据描述方法约定

符号	例子	说明
0x	0x0200，0x79	地址或数据以 16 进制表示。
0b	0b010，0b00 000 111	数据采用二进制表示 (寄存器描述除外)。
X	00X，XX1	数据描述中，X 代表 0 或 1。 例如，00X 代表 000 或 001；XX1 代表 001，011，101 或 111。

3 数值单位约定

本文档在描述数据容量（如 NAND 容量）时，单位词头代表的是 1024 的倍数；描述频率、数据速率等时则代表的是 1000 的倍数。具体如下：

表 1-3: 数值单位约定

类型	符号	对应数值
数据容量（如 NAND 容	1 K	1024
	1 M	1 048 576
	1 G	1 073 741 824
频率，数据速率等	1 K	1000
	1 M	1 000 000
	1 G	1 000 000 000

版权所有 © 珠海全志科技股份有限公司。保留一切权利

相关术语介绍

1.5.1 硬件术语

表 1-4: 硬件术语

术语	解释说明
CCU	时钟控制单元 (Clock Controll Unit)，也即时钟控制器
PLL	锁相环 (Phase Locked Loop)，利用输入信号和反馈信号的相位差异提高时钟频率

1.5.2 软件术语

表 1-5: 软件术语

术语	解释说明
UCF	全志实现的用于 RTOS 环境和裸机环境的通用时钟框架 (Universal Clock Framework)
K	Clock（时钟）的简称
HAL	Hardware Abstraction Layer，硬件抽象层

版权所有 © 珠海全志科技股份有限公司。保留一切权利

2 模块介绍

2.1 时钟控制器和时钟硬件

全志平台的时钟控制器，不仅有时钟硬件的控制寄存器，也有复位硬件的控制寄存器，因此软件层面的时钟框架实际上不仅管理时钟资源，也管理复位资源。

2.2 时钟硬件介绍

常见时钟硬件如下：

- 振荡器 (Oscillator)：所有时钟的源头，比如晶体振荡器、RC 振荡器等
- 门控 (Gate)：用于时钟信号的开启与关闭，一般基于与门实现
- 倍频器 (Multiplier，简称为 MULT)：用于提高时钟频率，一般基于锁相环 (PLL) 实现
- 分频器 (Divider，简称为 DIV)：用于降低时钟频率，一般基于计数器实现
- 选择器 (Multiplexer，简称为 MUX)：用于选择时钟来源 (父时钟)
- 其他处理时钟信号的复杂时钟硬件，比如 RCCAL，用于将输入时钟精准的分频到某个输出频率 (校准)

2.3 模块配置

2.3.1 menuconfig 配置说明

```
Drivers Options  --->
  Build Drivers Version (hal v2 version)  --->

  CCU Driver --->
    [*] Enable Clock Controller Unit Driver(Based On Universal Clock Framework)
    [ ]  enable CCU hal APIs test command
```

2.4 源码结构

1. 驱动源码

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
hal/source/ccu
├── clk_core
│   ├── aw_ccu.c
│   ├── aw_ccu.h
│   ├── ccu_div.c
│   ├── ccu_div.h
│   ├── ccu_gate.c
│   ├── ccu_gate.h
│   ├── ccu_multiplier.c
│   ├── ccu_multiplier.h
│   ├── ccu_mux.c
│   ├── ccu_mux.h
│   ├── ccu_pll.c
│   ├── ccu_pll.h
│   ├── ccu_reset.c
│   ├── ccu_reset.h
│   ├── clk_common.h
├── clk_core.c
│   ├── clk_core.h
│   ├── clk_fixed_factor.c
│   ├── clk_source.c
│   ├── clk_source.h
│   ├── include
│   │   ├── ucf_port.h
│   │   └── ucf_run_env.h
│   ├── Makefile
│   ├── port
│   │   ├── baremetal
│   │   │   └── baremetal_env_port.h
│   │   └── rtos
│   │       └── freertos
│   │           └── freertos_env_port.h
│   └── reset_common.h
├── hal_n_clk.c
├── hal_reset.c
├── Kconfig
├── Makefile
└── platform
    ├── Makefile
    ├── platform_ccu.h
    ├── platform_common.c
    ├── platform_common.h
    ├── platform_info.h
    ├── sun251iw1
    │   ├── Makefile
    │   ├── sun251iw1_app_ccu.c
    │   ├── sun251iw1_app_ccu.h
    │   ├── sun251iw1_ccu_id.h
    │   ├── sun251iw1_prcm_ccu.c
    │   ├── sun251iw1_prcm_ccu.h
    │   ├── sun251iw1_rtc_ccu.c
    │   ├── sun251iw1_rtc_ccu.h
    │   ├── sun251iw1_src_ccu.c
    │   └── sun251iw1_src_ccu.h
    ├── sun55iw6
    │   ├── Makefile
    │   ├── sun55iw6_ccu_id.h
    │   ├── sun55iw6_prcm_ccu.c
    │   ├── sun55iw6_prcm_ccu.h
    │   ├── sun55iw6_rtc_ccu.c
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
    ├── sun55iw6_rtc_ccu.h
    ├── sun55iw6_src_ccu.c
    ├── sun55iw6_src_ccu.h
    ├── sun55iw6_sys_ccu.c
    └── sun55iw6_sys_ccu.h
└── sun8iw22
    ├── Makefile
    ├── sun8iw22_ccu_id.h
    ├── sun8iw22_prcm_ccu.c
    ├── sun8iw22_prcm_ccu.h
    ├── sun8iw22_rtc_ccu.c
    ├── sun8iw22_rtc_ccu.h
    ├── sun8iw22_src_ccu.c
    ├── sun8iw22_src_ccu.h
    ├── sun8iw22_sys_ccu.c
    └── sun8iw22_sys_ccu.h
```

钟框架包含如下文件：

```
clk_core/*.c
hal_n_clk.c
hal_reset.c
```

具体平台的底层时钟驱动在 platform/xxx/目录，包含如下三类文件：

```
xxx_ccu_id.h /* 声明此平台时钟控制器ID */
xxx_yyy_ccu.h /* 声明某个时钟控制器包含的时钟ID和复位ID */
xxx_yyy_ccu.c /* 描述某个时钟控制器包含的时钟硬件的信息 */
```

比如 T153 平台对应的底层驱动文件如下：

```
platform/sun8iw22/sun8iw22_ccu_id.h
platform/sun8iw22/sun8iw22_prcm_ccu.c
platform/sun8iw22/sun8iw22_prcm_ccu.h
platform/sun8iw22/sun8iw22_rtc_ccu.c
platform/sun8iw22/sun8iw22_rtc_ccu.h
platform/sun8iw22/sun8iw22_src_ccu.c
platform/sun8iw22/sun8iw22_src_ccu.h
platform/sun8iw22/sun8iw22_sys_ccu.c
platform/sun8iw22/sun8iw22_sys_ccu.h
```

2. 驱动 APIs 声明头文件

```
include/hal/
└── hal_clk.h
```

明

hal_clk.h 是外设驱动或应用代码唯一可直接包含的头文件，不要去直接包含 hal/source/ccu 目录下的任何头文件！！！

3. 时钟、复位操作相关示例代码

```
hal/examples/ccu
├── clock_disable
│   └── clock_disable.c
├── clock_enable
│   └── clock_enable.c
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
├── clock_get_frequency
│   └── clock_get_frequency.c
├── clock_get_info
│   └── clock_get_info.c
├── clock_get_parent
│   └── clock_get_parent.c
├── clock_set_frequency
│   └── clock_set_frequency.c
├── clock_set_parent
│   └── clock_set_parent.c
├── reset_assert
│   └── reset_assert.c
├── reset_assert_bak
│   └── reset_assert.c
├── reset_deassert
│   └── reset_deassert.c
└── reset_trigger_reset
    └── reset_trigger_reset.c
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

3 模块接口说明

表 3-1: 模块接口列表

API	解释说明
n_clk_framework_init	(内部初始化使用，初始化时钟控制器驱动)
hal_n_clk_get_by_clk_num	通过时钟号获取时钟句柄
hal_n_clk_get	通过时钟控制器 ID 和时钟 ID 获取时钟句柄
hal_n_clk_put	释放时钟句柄
hal_n_clk_enable	打开指定时钟
hal_n_clk_disable	关闭指定时钟
hal_n_clk_get_enable_state	获取指定时钟的启用状态
n_clk_set_parent	设置指定时钟的父时钟
hal_n_clk_get_parent	获取指定时钟的父时钟
hal_n_clk_set_freq	设置指定时钟的频率
hal_n_clk_get_freq	获取指定时钟的频率
hal_n_clk_round_freq	获取某个时钟可成功设置的频率 (跟目标频率最接近的)
hal_n_clk_get_name	获取指定时钟的名称 (时钟名由底层时钟控制器驱动提供，命名不一定准确)
n_clk_get_clk_num	获取指定时钟的时钟号
hal_n_clk_get_id_info	获取指定时钟的时钟控制器 ID 和时钟 ID
hal_rst_get_by_rst_num	通过复位号获取复位句柄
hal_rst_get	通过复位控制器 ID 和复位 ID 获取复位句柄
hal_rst_put	释放复位句柄
hal_rst_deassert	让指定复位硬件退出复位状态
hal_rst_assert	让指定复位硬件进入复位状态

版权所有 © 珠海全志科技股份有限公司。保留一切权利

	解释说明
hal_rst_trigger_reset	让指定复位硬件触发一次完整的复位流程
hal_rst_get_status	获取指定复位硬件的状态

3.1 数据结构

3.1.1 clk_controller_id_t

述时钟控制器

```
typedef uint8_t clk_controller_id_t;
```

3.1.2 clk_id_t

用于描述时钟 ID：

```
typedef uint16_t clk_id_t;
```

3 clk_number_t

用于描述时钟号：

```
typedef uint32_t clk_number_t;
```

可使用 MAKE_CLKn 宏将时钟控制器 ID 和时钟 ID 转换成时钟号：

```
#define MAKE_CLKn(cc_id, clk_id) (((cc_id) << CLK_ID_BITS) | (clk_id))
```

3.1.4 hal_clk_t

用于描述时钟句柄：

```
typedef void * hal_clk_t;
```

3.1.5 rst_controller_id_t

用于描述复位控制器 ID：

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
typedef clk_controller_id_t rst_controller_id_t;
```

3.1.6 rst_id_t

用于描述复位 ID：

```
typedef uint16_t rst_id_t;
```

3.1.7 rst_number_t

述复位号：

```
typedef uint32_t clk_number_t;
```

可使用 MAKE_RSTn 宏将复位控制器 ID 和复位 ID 转换成复位号：

```
#define MAKE_RSTn(rc_id, rst_id) (((rc_id) << RST_ID_BITS) | (rst_id))
```

3.1.8 hal_rst_t

用于描述复位句柄：

```
typedef void *hal_rst_t;
```

3.2 API

3.2.1 hal_n_clk_framework_init

```
int hal_n_clk_framework_init(void);
```

- 作用：初始化时钟框架 (内部有初始化底层时钟控制器驱动)

:

  - void

- 返回：

  - void

版权所有 © 珠海全志科技股份有限公司。保留一切权利

3.2.2 hal_n_clk_get_by_clk_num

```
int hal_n_clk_get_by_clk_num(clk_number_t clk_num, hal_clk_t *clk);
```

- 作用：通过时钟号获取时钟句柄
- 参数：
  - clk_num：时钟号，可通过 MAKE_CLKn 宏来生成
  - clk：获取到的时钟句柄
- 返回：
  - int：返回值，0 表示执行成功，非 0 表示执行出错

3.2.3 hal_n_clk_get

```
int hal_n_clk_get(clk_controller_id_t cc_id, clk_id_t clk_id, hal_clk_t *clk);
```

- 作用：通过时钟控制器 ID 和时钟 ID 获取时钟句柄
- 参数：
  - cc_id：时钟控制器 ID
  - clk_id：时钟 ID
  - clk：获取到的时钟句柄
- 返回：
  - int：返回值，0 表示执行成功，非 0 表示执行出错

3.2.4 hal_n_clk_put

```
int hal_n_clk_put(hal_clk_t clk);
```

- 作用：释放时钟句柄
  - clk：时钟句柄
- 返回：
  - int：返回值，0 表示执行成功，非 0 表示执行出错

3.2.5 hal_n_clk_enable

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
int hal_n_clk_enable(hal_clk_t clk)
```

- 作用：打开指定时钟
- 参数：
  - clk：时钟句柄
- 返回：
  - int：返回值，0 表示执行成功，非 0 表示执行出错

3.2.6 hal_n_clk_disable

```
int hal_n_clk_disable(hal_clk_t clk)
```

- 作用：关闭指定时钟
- 参数：
  - clk：时钟句柄
- 返回：
  - int：返回值，0 表示执行成功，非 0 表示执行出错

3.2.7 hal_n_clk_get_enable_state

```
int hal_n_clk_get_enable_state(hal_clk_t clk, int *is_enabled)
```

- 作用：获取指定时钟的启用状态
- 参数：
  - clk：时钟句柄
  - is_enabled：对应时钟的启用状态，0 表示未启用，非 0 表示已启用
- 返回：
  - int：返回值，0 表示执行成功，非 0 表示执行出错

3.2.8 hal_n_clk_set_parent

```
int hal_n_clk_set_parent(hal_clk_t clk, hal_clk_t parent)
```

- 作用：设置指定时钟的父时钟

版权所有 © 珠海全志科技股份有限公司。保留一切权利

- 参数：
  - clk：时钟句柄
  - parent：父时钟的时钟句柄
- 返回：
  - int：返回值，0 表示执行成功，非 0 表示执行出错

3.2.9 hal_n_clk_get_parent

```
int hal_n_clk_get_parent(hal_clk_t clk, hal_clk_t *parent)
```

- 作用：获取指定时钟的父时钟
- 参数：
  - clk：时钟句柄
  - parent clk：获取到的父时钟的时钟句柄
- 返回：
  - int：返回值，0 表示执行成功，非 0 表示执行出错

10 hal_n_clk_set_freq

```
int hal_n_clk_set_freq(hal_clk_t clk, uint32_t freq)
```

- 作用：设置指定时钟的频率
- 参数：
  - clk：时钟句柄
  - freq：要设置的频率
- 返回：

执行成功，0 表示执行出错

3.2.11 hal_n_clk_get_freq

```
int hal_n_clk_get_freq(hal_clk_t clk, uint32_t *freq)
```

- 作用：获取指定时钟的频率
- 参数：

版权所有 © 珠海全志科技股份有限公司。保留一切权利

- clk：时钟句柄

freq：获取到的时钟频率

- 返回：
  - int：返回值，0 表示执行成功，非 0 表示执行出错

3.2.12 hal_n_clk_round_freq

```
int hal_n_clk_round_freq(hal_clk_t clk, uint32_t *freq)
```

：获取某个时钟可被设置的频率
：

- clk：时钟句柄
- freq：传入的目标频率，执行成功时保存获取到的可成功设置的频率

- 返回：
  - int：返回值，0 表示执行成功，非 0 表示执行出错

3.2.13 hal_n_clk_get_name

```
_n_clk_get_name(hal_clk_t clk, const char **name)
```

- 作用：获取指定时钟的名称 (时钟名由底层时钟控制器驱动提供，命名不一定准确)
- 参数：
  - clk：时钟句柄
  - name：获取到的时钟名
- 返回：
  - int：返回值，0 表示执行成功，非 0 表示执行出错

3.2.14 hal_n_clk_get_clk_num

```
int hal_n_clk_get_clk_num(hal_clk_t clk, clk_number_t *clk_num)
```

- 作用：获取指定时钟的时钟号
- 参数：
  - clk：时钟句柄

版权所有 © 珠海全志科技股份有限公司。保留一切权利

— clk_num：获取到的时钟号

— int：返回值，0 表示执行成功，非 0 表示执行出错

3.2.15 hal_n_clk_get_id_info

```
int hal_n_clk_get_id_info(hal_clk_t clk, clk_controller_id_t *cc_id, clk_id_t *clk_id)
```

- 作用：获取指定时钟的时钟控制器 ID 和时钟 ID
- 参数：
  - clk：时钟句柄
  - cc_id：获取到的时钟控制器 ID
  - clk_id：获取到的时钟 ID
- 返回：
  - int：返回值，0 表示执行成功，非 0 表示执行出错

3.2.16 hal_rst_get_by_rst_num

```
_rst_get_by_rst_num(rst_number_t rst_num, hal_rst_t *rst);
```

- 作用：通过复位号获取复位句柄
- 参数：
  - rst_num：复位号，可通过 MAKE_RSTn 宏来生成
  - rst：获取到的复位句柄
- 返回：
  - int：返回值，0 表示执行成功，非 0 表示执行出错

3.2.17 hal_rst_get

```
int hal_rst_get(rst_controller_id_t rc_id, rst_id_t rst_id, hal_rst_t *rst);
```

- 作用：通过复位控制器 ID 和复位 ID 获取复位句柄
- 参数：
  - rc_id：复位控制器 ID

版权所有 © 珠海全志科技股份有限公司。保留一切权利

  - rst_id：复位 ID

rst：获取到的复位句柄

- 返回：
  - int：返回值，0 表示执行成功，非 0 表示执行出错

3.2.18 hal_rst_put

```
int (hal_rst_t rst);
```

- 作用：释放复位句柄

：

  - rst：复位句柄
- 返回：
  - int：返回值，0 表示执行成功，非 0 表示执行出错

3.2.19 hal_rst_deassert

```
int hal_rst_deassert(hal_rst_t rst);
```

- 作用：让指定复位硬件退出复位状态
- 参数：
  - rst：复位句柄
- 返回：
  - int：返回值，0 表示执行成功，非 0 表示执行出错

3.2.20 hal_rst_assert

```
int hal_rst_assert(hal_rst_t rst);
```

- 作用：让指定复位硬件进入复位状态
- 参数：
  - rst：复位句柄
- 返回：
  - int：返回值，0 表示执行成功，非 0 表示执行出错

版权所有 © 珠海全志科技股份有限公司。保留一切权利

3.2.21 hal_rst_trigger_reset

```
int hal_rst_trigger_reset(hal_rst_t rst, uint32_t us);
```

- 作用：让指定复位硬件触发一次完整的复位流程
- 参数：
  - rst：复位句柄
  - us：复位硬件进入复位状态的保持时间
- 返回：

执行成功返回值，0 表示执行出错

3.2.22 hal_rst_get_status

```
int hal_rst_get_status(hal_rst_t rst, int *status);
```

- 作用：获取指定复位硬件的状态
- 参数：
  - rst：复位句柄
  - status：对应复位硬件的复位状态，非
- 返回：
  - int：返回值，0 表示执行成功，非 0 表示执行出错

版权所有 © 珠海全志科技股份有限公司。保留一切权利

4 功能开发

4.1 开发流程

步骤 1 确定目标时钟的时钟控制器 ID 和时钟 ID

调用 hal获取对应的clk_num

步骤 3 调用其他可接收句柄参数的 API 对时钟做具体操作 (开启、关闭、设置父时钟、设置频率等)

步骤 4 调用 hal_n_clk_put 释放时钟句柄。

说明

对于复位，也是类似按类似步骤操作即可。

4.1.1 确定某个时钟对应的时钟控制器 ID 和时钟 ID

由于时钟控制器 ID 和时钟 ID 是软件上定义的，无法从芯片 User Manual 里找到，因此需要在底层驱动里去找到具体平台上定义的这些

目前有 2 种方法去寻找时钟控制器 ID 和时钟 ID：

1. 方法 1

在对应平台的底层驱动的 xxx_yyy_ccu.h 里搜索关键字，下面以 T153 为例，查找 UART0 相关的时钟：

(1) T153 对应的平台代号为 sun8iw22，则其底层时钟驱动的相关文件在 hal/source/ccu/platform-sun8iw22.h 可以找到UART0的BUS_UART0

```
./sun8iw22_sys_ccu.h:158:     CLK_BUS_UART0,
```

(2) 在此目录下的同名的 c 文件 (sun8iw22_sys_ccu.c) 里查找 clk_controler_t 类型的变量定义，如下：

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
sun8iw22_sys_ccu.c:1286:clk_controller_t g_sys_ccu = {
sun8iw22_sys_ccu.c-1287-      .id = AW_SYS_CCU,
sun8iw22_sys_ccu.c-1288- .reg_base = SYS_CCU_REG_BASE,
sun8iw22_sys_ccu.c-1289-
sun8iw22_sys_ccu.c-1290-      .clk_num = ARRAY_SIZE(sun8iw22_sys_clk_hws),
sun8iw22_sys_ccu.c-1291-      .clk_hws = sun8iw22_sys_clk_hws,
```

可以看到其时钟控制器 ID 赋值为 AW_SYS_CCU。

(3) 因此 UART0 的时钟控制器 ID 为 AW_SYS_CCU，时钟 ID 为 CLK_BUS_UART0。

2. 方法 2

的 User manual 时钟相关章节，查找想要操作的时钟属于哪个时钟控制器以及对应的控制寄存器信息，再根据这些信息到对应平台的某个 CCU 相关驱动代码里查找时钟控制器 ID 和时钟 ID。

4.2 编程示例

4.2.1 启用时钟

示例代码如下 (截取自 hal/examples/ccu/clock_enable.c)：

```
mple_clock_enable(void)
{
    int ret;
    hal_clk_t clk;
    clk_number_t clk_num;

    clk_num = MAKE_CLKn(AW_SYS_CCU, CLK_APP_UART0_BUS);

    printf("clk info, clk_num: %u, cc_id: %u, clk_id: %u\n", clk_num, CC_ID(clk_num), CLK_ID(clk_num));

    ret = hal_n_clk_get_by_clk_num(clk_num, &clk);
    if (ret) {
        printf("get clk(%u) failed, ret: %d\n", clk_num, ret);
        return -1;
    }

    ret = hal_n_clk_enable(clk);
    if (ret) {
        printf("enable clk(%u) failed, ret: %d\n", clk_num, ret);
        ret = -2;
        goto exit_with_put_clk;
    }

    printf("enable clk(%u) success\n", clk_num);

    ret = 0;

exit_with_put_clk:
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
    hal_n_clk_put(clk);
    return ret;
```

更多编程示例请参考 hal/examples/ccu 目录下各时钟、复位操作的示例代码文件。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

5 调试方法

5.1 调试信息

CCU 驱动主要是管理和配置时钟，调试主要是检查是否能够正确设置时钟和获取时钟信息。

1. 查阅 User manual，查找目标时钟对应的控制寄存器信息，通过 p 命令读取控制器寄存器确认相关寄存器的值是否复合预期。
2. 通过 clk_dump 命令查看时钟树信息，下面是在 T153 平台上执行 clk_dump 命令输出的时钟树信息：

```
uart>clk_dump
     clk_num|name            |en_cnt| frequency|      parent
    0(00:000) hosc               0   24000000   NONE
    1(00:001) losc               0      32768   NONE
    2(00:002) rc_hf              0   16000000   NONE
 65536(01:000) pll-peri0             4 2400000000     0(00:000)
 65537(01:001) pll-peri0-2x          3 1200000000  65536(01:000)
 65538(01:002) pll-peri0-800m        1  800000000  65536(01:000)
(01:003) pll-peri0-480m 4 480000000 65536(01:000)
 65540(01:004) pll-peri0-600m        2  600000000  65537(01:001)
 65541(01:005) pll-peri0-400m        2  400000000  65537(01:001)
 65542(01:006) pll-peri0-300m        2  300000000  65540(01:004)
 65543(01:007) pll-peri0-200m        1  200000000  65541(01:005)
 65544(01:008) pll-peri0-160m        1  160000000  65539(01:003)
 65545(01:009) pll-peri0-150m        1  150000000  65542(01:006)
 65546(01:010) pll-48m               2   48000000  65539(01:003)
 65547(01:011) pll-48m-div-12m       1   12000000  65546(01:010)
 65548(01:012) pll-hosc-div-12m      1   12000000     0(00:000)
 65549(01:013) pll-peri1             0 2400000000     0(00:000)
 65550(01:014) pll-peri1-2x          0 1200000000  65549(01:013)
 65551(01:015) pll-peri1-800m        0  800000000  65549(01:013)
 65552(01:016) pll-peri1-480m        0  480000000  65549(01:013)
 65553(01:017) pll-peri1-600m        0  600000000  65550(01:014)
 65554(01:018) pll-peri1-400m        0  400000000  65550(01:014)
 65555(01:019) pll-peri1-300m        0  300000000  65553(01:017)
 65556(01:020) pll-peri1-200m        0  200000000  65554(01:018)
 65557(01:021) pll-peri1-160m        0  160000000  65552(01:016)
 65558(01:022) pll-peri1-150m        0  150000000  65555(01:019)
 65559(01:023) pll-video0            0 2376000000     0(00:000)
 65560(01:024) pll-video0-4x         0 1188000000  65559(01:023)
 65561(01:025) pll-video0-3x         0  792000000  65559(01:023)
 65562(01:026) pll-audio0-parent     0 3072000000     0(00:000)
 65563(01:027) pll-audio0            0  614400000  65562(01:026)
 65564(01:028) ahb                   0  200000000  65540(01:004)
 65565(01:029) apb0                  0  100000000  65540(01:004)
 65566(01:030) apb1                  0   24000000     0(00:000)
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
65567(01:031) apb-uart              1  24000000     0(00:000)
65568(01:032) mbus                  2  480000000  65539(01:003)
65569(01:033) stby-sys-peri0pll-gate     0  200000000  65564(01:028)
65570(01:034) spif-ahb-gate          0  200000000  65564(01:028)
65571(01:035) usb2p0-sys-ahb-gate      0  200000000  65564(01:028)
65572(01:036) gmac2-ahb-gate         0  200000000  65564(01:028)
65573(01:037) gmac1-ahb-gate         0  200000000  65564(01:028)
65574(01:038) gmac0-ahb-gate         0  200000000  65564(01:028)
65575(01:039) mcu-sys-ahb-gate       0  200000000  65564(01:028)
65576(01:040) smhc3-ahb-gate-sw      0  200000000  65564(01:028)
65577(01:041) smhc2-ahb-gate-sw      0  200000000  65564(01:028)
65578(01:042) smhc1-ahb-gate-sw      0  200000000  65564(01:028)
65579(01:043) smhc0-ahb-gate-sw      0  200000000  65564(01:028)
65580(01:044) video-out0-ahb-gate    0  200000000  65564(01:028)
65581(01:045) peri-apb0-gate         0  100000000  65565(01:029)
65582(01:046) peri-ahb-gate          0  200000000  65564(01:028)
65583(01:047) gmac2-mbus-bus         0  480000000  65568(01:032)
(01:048) gmac1-mbus-bus 0 480000000 65568(01:032)
65585(01:049) gmac0-mbus-bus         0  480000000  65568(01:032)
65586(01:050) isp-mbus-bus           0  480000000  65568(01:032)
65587(01:051) csi-mbus-bus           0  480000000  65568(01:032)
65588(01:052) dma1-mbus-bus          1  480000000  65568(01:032)
65589(01:053) ce-sys-mbus-bus        0  480000000  65568(01:032)
65590(01:054) dma0-mbus-bus          0  480000000  65568(01:032)
65591(01:055) dma1-mbus-sw-cfg-bus     0  480000000  65568(01:032)
65592(01:056) dma0-mbus-sw-cfg-bus     0  480000000  65568(01:032)
65593(01:057) lbc-mbus-sw-cfg-bus      0  480000000  65568(01:032)
65594(01:058) video-out0-mbus-sw-bus     0  480000000  65568(01:032)
65595(01:059) mcu-sys-mbus-sw-cfg-bus     0  480000000  65568(01:032)
65596(01:060) video-in-mbus-sw-bus     0  480000000  65568(01:032)
65597(01:061) can-mbus-sw-cfg-bus      0  480000000  65568(01:032)
65598(01:062) ce-sys-mbus-sw-cfg-bus     0  480000000  65568(01:032)
65599(01:063) gmac2-mbus-sw-cfg-bus     0  480000000  65568(01:032)
(01:064) gmac1-mbus-sw-cfg-bus 0 480000000 65568(01:032)
65601(01:065) gmac0-mbus-sw-cfg-bus     0  480000000  65568(01:032)
65602(01:066) gmac-mbus-sw-cfg-bus     0  480000000  65568(01:032)
65603(01:067) dma0                  0  24000000     0(00:000)
65604(01:068) dma1                  1  24000000     0(00:000)
65605(01:069) spinlock              0  24000000     0(00:000)
65606(01:070) msgbox0               0  24000000     0(00:000)
65607(01:071) msgbox-core0          0  24000000     0(00:000)
65608(01:072) msgbox-core1          0  24000000     0(00:000)
65609(01:073) msgbox-core2          0  24000000     0(00:000)
65610(01:074) msgbox-core3          0  24000000     0(00:000)
65611(01:075) msgbox-rv             0  24000000     0(00:000)
65612(01:076) pwm0                  0  24000000     0(00:000)
65613(01:077) pwm1                  0  24000000     0(00:000)
65614(01:078) pwm2                  0  24000000     0(00:000)
65615(01:079) dbgsys                0  24000000     0(00:000)
65616(01:080) sysdap                0  24000000     0(00:000)
65617(01:081) pmwcs0                0  24000000     0(00:000)
65618(01:082) pwmcs0_bus            0  24000000     0(00:000)
65619(01:083) pmwcs1                0  24000000     0(00:000)
65620(01:084) pwmcs1_bus            0  24000000     0(00:000)
65621(01:085) timer0-clk            0   6000000     0(00:000)
65622(01:086) timer1-clk            0  24000000     0(00:000)
65623(01:087) timer2-clk            0  24000000     0(00:000)
65624(01:088) timer3-clk            0  24000000     0(00:000)
65625(01:089) timer4-clk            0  24000000     0(00:000)
65626(01:090) timer5-clk            0  24000000     0(00:000)
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
65627(01:091) timer6-clk          0  24000000     0(00:000)
65628(01:092) timer7-clk          0  24000000     0(00:000)
65629(01:093) timer0-bus          0  24000000     0(00:000)
65630(01:094) timer0-0-riscv      0   6000000     0(00:000)
65631(01:095) timer0-1-riscv      0  24000000     0(00:000)
65632(01:096) timer0-2-riscv      0  24000000     0(00:000)
65633(01:097) timer0-3-riscv      0  24000000     0(00:000)
65634(01:098) timer0-riscv-bus    0  24000000     0(00:000)
65635(01:099) de0                 0  600000000  65540(01:004)
65636(01:100) de0-bus             0  24000000     0(00:000)
65637(01:101) g2d                 0  300000000  65540(01:004)
65638(01:102) g2d-bus             0  24000000     0(00:000)
65639(01:103) ce-sys              0  480000000  65539(01:003)
65640(01:104) ce-sys-bus          0  24000000     0(00:000)
65641(01:105) riscv-core          0  600000000  65540(01:004)
65642(01:106) riscv-ts            0  24000000     0(00:000)
65643(01:107) riscv-cfg-gate      0  24000000     0(00:000)
(01:108) dramc-ahb-gate 0 24000000 0(00:000)
65645(01:109) smhc0               0  13333333  65541(01:005)
65646(01:110) smhc0-ahb-gate      0  24000000     0(00:000)
65647(01:111) smhc1               0  300000000  65542(01:006)
65648(01:112) smhc1-ahb-gate      0  24000000     0(00:000)
65649(01:113) smhc2               0  800000000  65538(01:002)
65650(01:114) smhc2-ahb-gate      0  24000000     0(00:000)
65651(01:115) smhc3               0  400000000  65541(01:005)
65652(01:116) smhc3-ahb-gate      0  24000000     0(00:000)
65653(01:117) uart0-bus           0  24000000  65567(01:031)
65654(01:118) uart1-bus           0  24000000  65567(01:031)
65655(01:119) uart2-bus           0  24000000  65567(01:031)
65656(01:120) uart3-bus           0  24000000  65567(01:031)
65657(01:121) uart4-bus           0  24000000  65567(01:031)
65658(01:122) uart5-bus           0  24000000  65567(01:031)
65659(01:123) uart6-bus           0  24000000  65567(01:031)
24000000 65567(01:031)
65661(01:125) uart8-bus           0  24000000  65567(01:031)
65662(01:126) uart9-bus           0  24000000  65567(01:031)
65663(01:127) twi0-bus            0  24000000  65566(01:030)
65664(01:128) twi1-bus            0  24000000  65566(01:030)
65665(01:129) twi2-bus            0  24000000  65566(01:030)
65666(01:130) twi3-bus            0  24000000  65566(01:030)
65667(01:131) twi4-bus            0  24000000  65566(01:030)
65668(01:132) twi5-bus            0  24000000  65566(01:030)
65669(01:133) spi0                0  100000000  65542(01:006)
65670(01:134) spi0-bus            0  24000000     0(00:000)
65671(01:135) spi1                0  24000000     0(00:000)
65672(01:136) spi1-bus            0  24000000     0(00:000)
65673(01:137) spi2                0  24000000     0(00:000)
65674(01:138) spi2-bus            0  24000000     0(00:000)
65675(01:139) spif                0  24000000     0(00:000)
65676(01:140) spif-bus            0  24000000     0(00:000)
65677(01:141) spi3                0  100000000  65542(01:006)
65678(01:142) spi3-bus            0  24000000     0(00:000)
65679(01:143) gpadc0              0  24000000     0(00:000)
65680(01:144) gpadc0-bus          0  24000000     0(00:000)
65681(01:145) gpadc1              0  24000000     0(00:000)
65682(01:146) gpadc1-bus          0  24000000     0(00:000)
65683(01:147) gpadc2              0  24000000     0(00:000)
65684(01:148) gpadc2-bus          0  24000000     0(00:000)
65685(01:149) gpadc3              0  24000000     0(00:000)
65686(01:150) gpadc3-bus          0  24000000     0(00:000)
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
65687(01:151) tsensor-bus          0  24000000     0(00:000)
65688(01:152) ir_rx0               0  24000000     0(00:000)
65689(01:153) ir-rx0-bus           0  24000000     0(00:000)
65690(01:154) ir_tx0               0  12000000     0(00:000)
65691(01:155) ir-tx0-bus           0  24000000     0(00:000)
65692(01:156) tpadc                0  24000000     0(00:000)
65693(01:157) tpadc-bus            0  24000000     0(00:000)
65694(01:158) lbc                  0 480000000 65539(01:003)
65695(01:159) lbc-bus              0  24000000     0(00:000)
65696(01:160) ir_rx1               0     32768     1(00:001)
65697(01:161) ir-rx1-bus           0  24000000     0(00:000)
65698(01:162) ir_rx2               0     32768     1(00:001)
65699(01:163) ir-rx2-bus           0  24000000     0(00:000)
65700(01:164) ir_rx3               0     32768     1(00:001)
65701(01:165) ir-rx3-bus           0  24000000     0(00:000)
65702(01:166) i2s0                 0 614400000 65563(01:027)
65703(01:167) i2s0-bus             0  24000000     0(00:000)
(01:168) 65563(01:027)
65705(01:169) i2s1-bus             0  24000000     0(00:000)
65706(01:170) i2s2                 0 614400000 65563(01:027)
65707(01:171) i2s2-bus             0  24000000     0(00:000)
65708(01:172) owa0-tx              0 614400000 65563(01:027)
65709(01:173) owa0-rx              0 400000000 65541(01:005)
65710(01:174) owa0-bus             0  24000000     0(00:000)
65711(01:175) dmic                 0 614400000 65563(01:027)
65712(01:176) dmic-bus             0  24000000     0(00:000)
65713(01:177) audiocodec0-dac      0 614400000 65563(01:027)
65714(01:178) audiocodec0-dac-bus  0  24000000     0(00:000)
65715(01:179) usb0                 0  12000000 65547(01:011)
65716(01:180) usb0-dev-bus         0  24000000     0(00:000)
65717(01:181) usb0-ehci-bus        0  24000000     0(00:000)
65718(01:182) usb0-ohci-bus        0  24000000     0(00:000)
65719(01:183) usb1                 0  12000000 65547(01:011)
(01:184) usb1-ehci-bus 0 24000000 0(00:000)
65721(01:185) usb1-ohci-bus        0  24000000     0(00:000)
65722(01:186) usb2p0-sys-phy       0  24000000     0(00:000)
65723(01:187) usb2p0-sys-bus       0  24000000     0(00:000)
65724(01:188) gmac0-phy            0  24000000     0(00:000)
65725(01:189) gmac0-ptp-ref        0  24000000     0(00:000)
65726(01:190) gmac0-bus            0  24000000     0(00:000)
65727(01:191) gmac1-phy            0  24000000     0(00:000)
65728(01:192) gmac1-ptp-ref        0  24000000     0(00:000)
65729(01:193) gmac1-bus            0  24000000     0(00:000)
65730(01:194) gmac2-phy            0  24000000     0(00:000)
65731(01:195) gmac2-ptp-ref        0  24000000     0(00:000)
65732(01:196) gmac2-bus            0  24000000     0(00:000)
65733(01:197) tcon-lcd0            0 1188000000 65560(01:024)
65734(01:198) tcon-lcd0-bus        0  24000000     0(00:000)
65735(01:199) mipi-dsi0            0  24000000     0(00:000)
65736(01:200) mipi-dsi0-bus        0  24000000     0(00:000)
65737(01:201) combophy0            0 1188000000 65560(01:024)
65738(01:202) vo0-reg-bus          0  24000000     0(00:000)
65739(01:203) ledc                 0  24000000     0(00:000)
65740(01:204) ledc-bus             0  24000000     0(00:000)
65741(01:205) csi-master0          0  24000000     0(00:000)
65742(01:206) csi-master1          0  24000000     0(00:000)
65743(01:207) csi-master2          0  24000000     0(00:000)
65744(01:208) csi                  0 200000000 65541(01:005)
65745(01:209) isp                  0 160000000 65539(01:003)
65746(01:210) video-in-bus         0  24000000     0(00:000)
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
131072(02:000) ahbs             0  200000000  65556(01:020)
131073(02:001) apbs0            0   24000000     0(00:000)
131074(02:002) apbs1            0   24000000     0(00:000)
131075(02:003) twd-bus          0   24000000     0(00:000)
131076(02:004) rtc-bus          0   24000000     0(00:000)
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

声明

版权所有 ©2025 珠海全志科技股份有限公司。保留一切权利。

本文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。

本文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、发表或传播本文档内容的部分或全部，且不得以任何形式传播。

商标声明

、 、 、 （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。在本文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。本文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，并严格遵循本文档的使用说明。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。

本文档作为使用指导仅供参考。由于产品版本升级或其他原因，本文档内容有可能修改，如有变不另行通知。全志尽全力在本文档中提供准确的信息，但并不确保内容完全没有错误，因使用本文档而发生损害（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。本文档中的所有陈述、信息和建议并不构成任何明示或暗示的保证或承诺。

本文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。