

# HPMicro 高性能 MCU

## HPM SDK 使用指南

1 简介	. 4
2 开发前的准备	. 5
2.1 硬件环境准备	5
2.2 软件环境准备	7
2.3 开发调试环境搭建	. 8
3 HPM SDK 介绍	8
3.1 SDK ENV 组成结构	. 9
3.2 HPM SDK 核心软件包架构	10
4 HPM SDK 驱动模块简介	11
4.1 HPM SDK 驱动文件结构	11
4.2 HPM SDK 驱动 API	13
4.3 HPM SDK 数据结构	15
4.4 HPM SDK 宏定义	17
4.5 HPM SDK 函数接口定义	17
4.6 HPM SDK API 应用示例	18
6 HPM SDK 中间件	19
7 HPM SDK 应用例程	19
8 HPM SDK 常见问题	25



版本:

日期	版本号	说明
2022-4-25	1.0	初版

## 1 简介

本文将为先楫半导体的高性能 MCU 配套的 HPM SDK (以下可简称 SDK) 提供基本的说明,为用户提供 HPM SDK 入门指导。

HPM SDK 是一个基于宽松使用许可(BSD 3-Clause)完全开源的综合性 软件支持包,可帮助用户在使用基于 RISC-V 内核的先楫半导体 MCU 时能简化 和加快应用开发。

HPM SDK 包括第三方工具软件、硬件驱动软件、集成实时操作系统、中间件、参考例程以及说明文档等。HPM SDK 支持用 Cmake 为 Segger 和 GCC 提供示例项目,并同时集成进 RT Thread、Zephyr 等开源项目中。

用户在先楫半导体官网平台选择所需要的 MCU、评估板和可选组件后即可 下载获得对应的 SDK 支持包。

HPM SDK 主要由以下组件组成:

- RISC-V 启动文件和设备头文件及标准库。
- 开源外设驱动程序提供无状态、高性能、易于使用的 API。
- 通信外设的驱动程序还包括用于高性能数据传输的高级通信 API。
- 预集成的实时操作系统(RTOS)内核: FreeRTOS。
- 先楫半导体和合作伙伴提供的软件支持组件,包括:
  - 图形和人机界面
  - 音频和语音处理
  - 电机控制
  - 信息安全
  - 存储
  - 无线和有线通信
- 展示外设驱动程序、实时操作系统包驱动程序、中间件和 RTOS 的用法 的示例软件



## 2 开发前的准备

### 2.1 硬件环境准备

(1)HPM67xx系列 MCU 的开发板:HPM6750 EVK开发板和HPM6750 EVK mini 开发板。

HPM6750EVK 开发板如下图:



HPM6750EVK 板上硬件资源:

- HPM6750IVM 微控制器 (主频 816Mhz, 2MB 片上内存)
- 板载存储
  - 256Mb SDRAM
  - 128Mb Quad SPI NOR Flash
- 显示/摄像头
  - LCD 接口
  - 摄像头(DVP)接口
- 以太网
  - 1000 Mbits PHY
  - 100 Mbits PHY
- USB
  - USB type C (USB 2.0 OTG) connector x3





- 音频
  - Line in
  - Mic
  - Speaker
  - DAO
- 其他
  - TF 卡槽
  - FT2232
  - 蜂鸣器
  - RGB LED
  - CAN
- 扩展口
  - 电机控制

HPM6750 EVK mini 开发板如下图:



HPM6750EVKmini 板上硬件资源:

- HPM6750IVM 微控制器 (主频 816Mhz, 2MB 片上内存)
- 板载存储
  - 128Mb SDRAM





- 64Mb Quad SPI NOR Flash
- 显示/摄像头
  - LCD 接口
  - 摄像头(DVP)接口
- WiFi
  - RW007
- USB
  - USB type C (USB 2.0 OTG) connector x2
- 音频
  - Mic
  - DAO
- 其他
  - TF 卡槽
  - FT2232
  - 蜂鸣器
  - RGB LED
- 扩展口
  - ART-PI

(2) 一条 USB-TypeC 线缆,用于给 HPM6750 EVK mini 开发板供电,以及 调试。

(3) 一台 PC 主机或者笔记本电脑。

### 2.2 软件环境准备

为了更好的进行 HPM 6700 系列 MCU 的开发,建议您应当至少具备以下开发环境:

- Windows 使用 Windows 7 以上版本。
- Linux 建议使用 Ubuntu 18 以上的 LTS 版本。
- 安装 Segger Embedded Studio,使用 RISCV\_v620a\_win\_x64 或者 以上版本。
- 运行 sdk\_env\tools\FTDI\_InstallDriver.exe 以安装可用于调试的 FT2232 驱动。

### 2.3 开发调试环境搭建

具体开发板的连接,请参考硬件连接指南部分,请确保开发板被正确设置后 再进行下面的步骤软件开发调试,以 HPM6750EVKMini 为例,开发环境连接 示意如下:



## 3 HPM SDK 介绍

HPM 软件开发工具包(HPM SDK)是一套用于先楫半导体微控制器的软件, 包括底层外设驱动程序、组件、中间件和集成的 RTOS 支持。除了这些组件之外, HPM SDK 还提供了相关的演示和示例,以及文档,以帮助用户评估 HPM 产品 并有效地构建他们的应用程序。

	Demos				Application			Driver Examples				Doc & Helper Tools	
	Middleware coremark fatfs					tinycr	ypt	ti	nyusb				
			oremark		FreeRTOS	hpm_	mcl	libjp	eg-turbo	little	evgl		Tool Chains
			fatfs	f	freemaster	hpm_r	nath	hpm	n_sdmmc	Iw	ip		
SDK		SDXC	SP	I	UART	USB	V	AD	WDG	SDP			
	Duiteren	GPIO	120	:	JPEG	PDM	P٧	vм	RNG	RTC			
	Drivers	ACMP	CAN	1	DAO	DRAM	PLI	.CTL	PMP	PMU	PTPC		BSP&Start Up Files
		ADC	CAN	1	DMA	ENET	GP	TMR	125	LCDC	PDMA		
	Hardware(HPM67xx)												

Hardware: 基于 RISC-V 内核的高性能 MCU HPM6750, 该芯片拥有最大 2M 字节的连续片上 RAM, 并集成了丰富的存储接口, 如 SDRAM, Quad SPI NOR

flash, SD/eMMC卡。同时它也提供多种音视频接口包括LCD显示,像素DMA, 摄像头以及 I2S 音频接口。

Drivers: 抽象访问、操作硬件寄存器的 C 函数驱动接口。

Middleware: 在驱动程序的基础上增加功能,例如通信栈、图形库、通用算法、操作系统等。

通过浏览器打开以下文件,可快速方便的查询 HPM SDK 的详细介绍。

" sdk\_env\_v0.10.0/hpm\_sdk/doc/output/sdk\_doc/zh/latest/html/doc/RE ADME zh.html"



### 3.1 SDK ENV 组成结构

整个 SDK ENV 组成结	訥如下图所示:
	sdk_env_v0.10.0 >
	名称 人名
	🚞 doc
	hpm_sdk
	📒 toolchains
	tools
	CHANGELOG.md
	📄 list
	README.md
	README_zh.md
	🐁 start_cmd

- **doc**: 主要是先楫半导体官方开发的用户指导手册, 包含 HPM6750EVK 和 HPM6750EVK mini 的用户指导手册。
- hpm\_sdk: HPM SDK 的核心软件包。

<b>HPMicro</b>							先	<u></u> 揖半	导体	
• toolchains	: 编	译	I	具	链	,	主	要	为	
rv32imadc-ilp	rv32imadc-ilp32d-x86_64-w64-mingw32。									
● tools:主要第三	E方软件和 sdk_env_v0	相应的 .10.0 >	)处理朋 tools	却本,	如下图	2:				
	名称 Cmake									
	openocd									
	📒 pytho	on3								
	📒 script	S								
	🛃 FTDI_	InstallDi	river							
● 其他: SDK 运行环境配置脚本和说明文件。										

### 3.2 HPM SDK 核心软件包架构

SDK 核心软件包主要是支持完成应用软件开发的各种组件,包含了驱动、 板级支持文件、中间件、soc 定义文件、实例、处理脚步、帮助文件等。如下图 所示

名称	
📜 arch	
📒 boards	
📒 cmake	
🚞 components	
📒 doc	
🚞 drivers	
📜 middleware	
🚞 samples	
🚞 scripts	
soc	
📜 utils	
CHANGELOG.md	
CMakeLists	
🕲 env	
🗋 env.sh	
🗋 index.md	

各个目录的主要内容:

● arch: 与 risc-v 架构相关的操作接口。

- boards: 与开发板相关的硬件定义和接口, 主要为 HPM6750 EVK 和 HPM6750 EVK mini 相关的板卡硬件信息。
- **cmake:** 与 cmake 和工程管理相关配置文件。
- components: 常用板载模块驱动、处理逻辑和控制接口,包括 camera、 codec、debug\_console、enet\_phy、serial\_nor、touch、usb 等。
- doc: SDK 支持文档和软件 API 的支持文档。
- **drivers:** soc 片上硬件模块驱动、处理逻辑和控制接口。
- middleware:常见算法库、图形库、组件库、RTOS等。
- samples: 各种应用例程。
- script: 与工程管理、编译相关的处理脚本。
- soc: soc 片上资源寄存器定义。
- utils: 其他通用处理组件。
- 其他:处理脚步和说明文件。

## 4 HPM SDK 驱动模块简介

### 4.1 HPM SDK 驱动文件结构

HPM SDK 的驱动模块目录结构如下图所示,包括了头文件目录 inc 和源代码 目录 src。

dr	rivers
1.  -	-inc
2.	hpm_acmp_drv.h
3.	hpm_adc12_drv.h
4.	hpm_adc16_drv.h
5.	hpm_bacc_drv.h
6.	hpm_bkey_drv.h
7.	hpm_butn_drv.h
8.	hpm_cam_drv.h
9.	hpm_can_drv.h
10.	hpm_common.h
11.	hpm_dao_drv.h
12.	hpm_display_common.h
13.	hpm_dmamux_drv.h
14.	hpm_dma_drv.h
15.	hpm_dram_drv.h
16.	hpm_enet_drv.h
17.	hpm_gpiom_drv.h

先楫	ド导体
----	-----

18.	hpm_gpio_drv.h
19.	hpm_gptmr_drv.h
20.	hpm_hall_drv.h
21.	hpm_i2c_drv.h
22.	hpm_i2s_common.h
23.	hpm_i2s_drv.h
24.	hpm_jpeg_drv.h
25.	hpm_lcdc_drv.h
26.	hpm_mbx_drv.h
27.	hpm_mchtmr_drv.h
28.	hpm_mono_drv.h
29.	hpm_pcfg_drv.h
30.	hpm_pdma_drv.h
31.	hpm_pdm_drv.h
32.	hpm_pllctl_drv.h
33.	hpm_pmon_drv.h
34.	hpm_pmp_drv.h
35.	hpm_ppor_drv.h
36.	hpm_psec_drv.h
37.	hpm_ptpc_drv.h
38.	hpm_pwm_drv.h
39.	hpm_qei_drv.h
40.	hpm_rng_drv.h
41.	hpm_romapi_xpi_def.h
42.	hpm_romapi_xpi_nor_def.h
43.	hpm_romapi_xpi_ram_def.h
44.	hpm_rtc_drv.h
45.	hpm_sdp_drv.h
46.	hpm_sdxc_drv.h
47.	hpm_spi_drv.h
48.	hpm_synt_drv.h
49.	hpm_trgm_drv.h
50.	hpm_uart_drv.h
51.	hpm_usb_drv.h
52.	hpm_vad_drv.h
53.	hpm_wdg_drv.h
54.	
55.	—src
56.	hpm_acmp_drv.c
57.	hpm_adc12_drv.c
58.	hpm_adc16_drv.c
59.	hpm_cam_drv.c
60.	hpm_can_drv.c
61.	hpm_dao_drv.c

62.	hpm_dma_drv.c
63.	hpm_dram_drv.c
64.	hpm_enet_drv.c
65.	hpm_gpio_drv.c
66.	hpm_gptmr_drv.c
67.	hpm_i2c_drv.c
68.	hpm_i2s_drv.c
69.	hpm_jpeg_drv.c
70.	hpm_lcdc_drv.c
71.	hpm_pdma_drv.c
72.	hpm_pdm_drv.c
73.	hpm_pllctl_drv.c
74.	hpm_pmp_drv.c
75.	hpm_pmu_drv.c
76.	hpm_ptpc_drv.c
77.	hpm_pwm_drv.c
78.	hpm_rng_drv.c
79.	hpm_rtc_drv.c
80.	hpm_sdp_drv.c
81.	hpm_sdxc_drv.c
82.	hpm_spi_drv.c
83.	hpm_uart_drv.c
84.	hpm_usb_drv.c
85.	hpm_vad_drv.c
86.	hpm_wdg_drv.c

soc 的外设(peripheral device)驱动源代码在"\hpm\_sdk\drivers\src"目 录下,每一个外设(如 CAN、SPI、UART 等)为一个独立的源文件,文件命名形 式:

#### hpm\_<*peripheral device*>\_drv.c

soc 的外设(peripheral device)驱动头文件在"\hpm\_sdk\drivers\inc"目 录下,每一个外设(如 CAN、SPI、UART 等)为一个独立的头文件,文件命名形 式:

hpm peripheral device> drv.h

### 4.2 HPM SDK 驱动 API

HPM SDK 提供了丰富的外设接口, 主要的外设 API 见下表:





ACMP driver APIs	LCD driver APIs
ADC12 driver APIs	MBX driver APIs
ADC16 driver APIs	MCHTMR driver APIs
BACC driver APIs	MONO driver APIs
BKEY driver APIs	OTP driver APIs
BUTN driver APIs	PDM driver APIs
CAM driver APIs	PDMA driver APIs
CAN driver APIs	PLIC driver APIs
CLOCK driver APIs	PLLCTL driver APIs
COMMON driver APIs	PTPC driver APIs
DAO driver APIs	PWM driver APIs
DMA driver APIs	QEI driver APIs
DMAMUX driver APIs	ROM APIs
DRAM driver APIs	RTC driver APIs
Display_common driver APIs	SDP driver APIs
Enet driver APIs	SDXC driver APIs
GPIO driver APIs	SPI driver APIs
GPIOM driver APIs	SYSCTL driver APIs
GPTMR driver APIs	TRGM driver APIs
HALL driver APIs	UART driver APIs
I2C driver APIs	USB driver APIs
I2S common driver APIs	VAD driver APIs
I2S driver APIs	WDG driver APIs
INTERRUPT driver APIs	XPI NOR driver APIs
JPEG driver APIs	XPI RAM driver APIs
L1CACHE driver APIs	XPI driver APIs

通过浏览器可以打开以下文件,方便查询 HPM SDK 中 API 相关的介绍。 "sdk\_env\_v0.10.0/hpm\_sdk/doc/output/api\_doc/html/index.html"



### 4.3 HPM SDK 数据结构

HPM SDK 中数据结构具备按照以下规则命名:

(1) 外设模块寄存器数据结构定义

外设模块名\_Type

比如 ADC 的寄存器定义如下:

```
1.
   typedef struct {
      __RW uint32_t CONFIG[12]; /* 0x0 - 0x2C: */
2.
      __RW uint32_t TRG_DMA_ADDR; /* 0x30: */
3.
      __R uint8_t RESERVED0[972]; /* 0x34 - 0x3FF: Reserved */
4.
      R uint32 t BUS RESULT[19]; /* 0x400 - 0x448: */
5.
      __R uint8_t RESERVED1[180]; /* 0x44C - 0x4FF: Reserved */
6.
7.
      RW uint32 t BUF CFG0; /* 0x500: */
8.
      __R uint8_t RESERVED2[764]; /* 0x504 - 0x7FF: Reserved */
9.
      __RW uint32_t SEQ_CFG0; /* 0x800: */
      ___RW uint32_t SEQ_DMA_ADDR; /* 0x804: */
10.
11.
      R uint32 t SEQ WR ADDR; /* 0x808: */
      RW uint32 t SEQ DMA CFG; /* 0x80C: */
12.
      __RW uint32_t SEQ_QUE[16]; /* 0x810 - 0x84C: */
13.
      ___R uint8_t RESERVED3[944]; /* 0x850 - 0xBFF: Reserved */
14.
15.
      struct {
      ___RW uint32_t PRD_CFG; /* 0xC00: */
16.
17.
        __RW uint32_t PRD_THSHD_CFG; /* 0xC04: */
18.
        __R uint32_t PRD_RESULT; /* 0xC08: */
19.
        __R uint8_t RESERVED0[4]; /* 0xC0C - 0xC0F: Reserved */
      } PRD_CFG[19];
20.
      R uint8 t RESERVED4[720]; /* 0xD30 - 0xFFF: Reserved */
21.
      ___RW uint32_t SAMPLE_CFG[19]; /* 0x1000 - 0x1048: */
22.
      __R uint8_t RESERVED5[184]; /* 0x104C - 0x1103: Reserved */
23.
24.
      __RW uint32_t CONV_CFG1; /* 0x1104: */
```





- \_\_RW uint32\_t ADC\_CFG0; /\* 0x1108: \*/ 25. 26. R uint8 t RESERVED6[4]; /\* 0x110C - 0x110F: Reserved \*/ 27. \_\_\_RW uint32\_t INT\_STS; /\* 0x1110: \*/ \_\_\_RW uint32\_t INT\_EN; /\* 0x1114: \*/ 28. 29. \_\_R uint8\_t RESERVED7[232]; /\* 0x1118 - 0x11FF: Reserved \*/ 30. \_\_\_RW uint32\_t ANA\_CTRL0; /\* 0x1200: \*/ \_\_\_RW uint32\_t ANA\_CTRL1; /\* 0x1204: \*/ 31. 32. \_\_R uint8\_t RESERVED8[8]; /\* 0x1208 - 0x120F: Reserved \*/ 33. \_\_\_RW uint32\_t ANA\_STATUS; /\* 0x1210: \*/

#### (2) 外设模块通用配置数据结构定义

外设模块名\_config\_t

比如 ADC 的通用配置数据结构定义如下:

- 1. typedef struct {
- uint8\_t ch;
- 3. uint8\_t diff\_sel;
- uint8\_t res;
- 5. uint8\_t sample\_cycle\_shift;
- 6. uint8\_t conv\_mode;
- 7. uint8\_t wait\_dis;
- uint16\_t thshdh;
- 9. uint16\_t thshdl;
- 10. uint32\_t sample\_cycle;
- 11. uint32\_t adc\_clk\_div;
- 12. bool sel\_sync\_ahb;
- 13. bool adc\_ahb\_en;
- 14. } adc12\_config\_t;

#### (3) 外设模块工作模式数据结构定义

#### 外设模块名 模式 config t

#### 比如 CAN 的 filter 配置数据结构定义如下:

- 1. typedef struct {
- 2. uint16\_t index;
- 3. can\_filter\_mode\_t mode;
- bool enable;
- 5. uint32\_t code;
- 6. uint32\_t mask;
- 7. } can\_filter\_config\_t;

各个外设模块的配置方式不同,对应的数据结构也会有相应的差异,具体各 个外设模块的数据结构定义与说明可以参考 SDK 的说明指导文档,文档路径: *"sdk\_env\_v0.10.0/hpm\_sdk/doc/output/api\_doc/html/annotated.html"* 注意:所有硬件外设相关的命名和数据手册的寄存器命名一致,地址偏移一致, 请勿随意修改。

### 4.4 HPM SDK 宏定义

(1) 外设模块工作模式数据结构定义

#define 外设模块名 操作 外设模块名 操作

#### 比如 CAN 的事件宏定义如下:

- 1. #define CAN\_EVENT\_RECEIVE (CAN\_RTIF\_RIF\_MASK)
- #define CAN\_EVENT\_RX\_BUF\_OVERRUN (CAN\_RTIF\_ROIF\_MASK)
- 3. #define CAN\_EVENT\_RX\_BUF\_FULL (CAN\_RTIF\_RFIF\_MASK)
- 4. #define CAN\_EVENT\_RX\_BUF\_ALMOST\_FULL (CAN\_RTIF\_RAFIF\_MASK)
- 5. #define CAN\_EVENT\_TX\_PRIMARY\_BUF (CAN\_RTIF\_TPIF\_MASK)
- 6. #define CAN\_EVENT\_TX\_SECONDARY\_BUF (CAN\_RTIF\_TSIF\_MASK)
- 7. #define CAN\_EVENT\_ERROR (CAN\_RTIF\_EIF\_MASK)
- 8. #define CAN\_EVENT\_ABORT (CAN\_RTIF\_AIF\_MASK )

### 4.5 HPM SDK 函数接口定义

HPM SDK 中函数的命名基本使用英文缩写来表达函数的完成功能, 遵循动 宾结构的命名法则, 函数名中动词在前,并在命名前加入函数的前缀, 外设模块 操作函数接口定义规则如下

函数返回值 外设模块名\_目标词\_动词\_宾语(函数参数)

比如 RTC 的函数定义如下:

RTC Functions
 hpm\_stat\_t rtc\_config\_time (RTC\_Type \*base, time\_t time)
 Configure the RTC time.
 hpm\_stat\_t rtc\_config\_alarm (RTC\_Type \*base, rtc\_alarm\_config\_t \*con fig)
 Configure RTC Alarm.
 Configure RTC Alarm.
 time\_t rtc\_get\_time (RTC\_Type \*base)
 Get the time returned by RTC module.
 10.



```
    static void rtc_enable_alarm_interrupt (RTC_Type *base, uint32_t inde x, bool enable)
    Enable RTC alarm interrupt.
    13.
    14. static void rtc_clear_alarm_flag (RTC_Type *base, uint32_t index)
    15. Clear RTC alarm flag.
    16.
    17. static bool rtc_is_alarm_flag_asserted (RTC_Type *base, uint32_t inde x)
    18. Check whether RTC alarm flag is set or not.
```

### 4.6 HPM SDK API 应用示例

下面以 PWM 的初始化为示例, 说明驱动模块的数据结构以及 API 的使

```
用方式:
```

```
1.
       pwm_cmp_config_t cmp_config = {0 };
2.
       pwm_config_t pwm_config = {0};
3.
4.
       pwm stop counter(ptr);
5.
       pwm_get_default_pwm_config(ptr, &pwm_config);
6.
       pwm_get_default_cmp_config(ptr, &cmp_config);
7.
8.
       pwm config.enable output = false;
9.
       pwm_config.dead_zone_in_half_cycle = 0;
10.
       pwm_config.invert_output = false;
11.
12.
       /* reload and start counter */
13.
       pwm set reload(ptr, 0, reload);
       pwm_set_start_count(ptr, 0, 0);
14.
15.
16.
       cmp_config.mode = pwm_cmp_mode_output_compare;
17.
       cmp_config.cmp = cmp_initial_zero ? 0 : reload + 1;
18.
       cmp_config.update_trigger = pwm_shadow_register_update_on_modify;
19.
20. /* config initial compare value which should take affect immediately
   */
21.
       pwm_config_cmp(ptr, cmp_index, &cmp_config);
22.
23. /* update trigger type so that compare value will be updated on hardw
   are event (RELOAD) */
       cmp_config.update_trigger = pwm_shadow_register_update_on_hw_even
24.
   t;
```





25.	/* *	' confi	g pwm as	outp	ut dri	.ven	by cmp	*/						
26.		if (st	atus_suc	cess	!= pwm	_set	up_wav	efor	m(ptr	, pir	n, &	owm_	config	, с
	mp_i	ndex,	&cmp_cor	fig,	1)) {									
27.		pr	intf("fa	iled	to set	up w	avefor	n∖n"	);					
28.		wh	ile(1);											
29.		}												
30.														
31.		/*	* confi	g hw	event		*/							
32.		cmp_cc	onfig.cmp	= re	load -	1;								
33.		cmp_cc	onfig.upd	late_t	rigger	• = p	wm_sha	dow_	regis	ter_u	ipdat	te_o	n_hw_e	ven
	t;													
34.		pwm lc	ad cmp s	hadow	on ma	tch(	ptr, h	√ ev	ent c	mp, 8	Scmp	con	fig);	

## 6 HPM SDK 中间件

HPM SDK 提供大部分通用的中间件, 支持用户更便捷的完成更多复杂的功能, HPM SDK 提供的中间如下:

	HP	M SDK中Ì	间件	
	tflm	tinycrypt	tinyusb	
coremark	FreeRTOS	hpm_mcl	libjpeg-turbo	littlevgl
fatfs	freemaster	hpm_math	hpm_sdmmc	lwip

## 7 HPM SDK 应用例程

HPM SDK 提供了丰富应用例程,帮助用户快速熟悉、掌握 SDK 内容,包括驱动、API、开发环境的使用方法。

例程	描述
audio_codec	音频解码例程
coremark	Coremark 例程,获取 CoreMark 性能评分
dhrystone	Dhrystone 例程,获取 Dhrystone 性能评分
drivers	包含了 SDK 所有驱动的应用参考例程
dsp	DSP 功能相关的应用例程
freemaster	嵌入式系统数据可视化工具 Freemaster 例程

hello_world	第一个 SDK 使用入门例程
jpeg	jpeg 功能相关的应用例程
littlevgl	littlevgl 功能相关的应用例程
lwip	Lwip 功能相关的应用例程
motor_ctrl	电机控制功能相关的应用例程
multicore	双核同步运行功能相关的应用例程
openocd_algo	openocd 功能相关的应用例程
rgb_led	RGB 流水灯功能相关的应用例程
rom_api	ROM 操作功能相关的应用例程
rtos	FreeRTOS 相关的应用例程
tflm	TensorflowLiteMicro 相关的应用例程
tinycrypt	TinyCrypt 加密工具相关的应用例程
tinyusb	tinyusb 功能相关的应用例程
touch_panel	触摸功能相关的应用例程

先楫半导体

HPM SDK 中各个例程的描述也可以在路径找到:

#### "hpm\_sdk/doc/output/sdk\_doc/zh/latest/html/samples/index\_zh.html"



下面以 SDK 中 "rgb\_led" 例程, 来示范 SDK 的应用例程的使用方法, 依

据《HPM6750EVKMINI\_UG》中的章节"3.4 sdk\_env/Segger Embedded Studio For RISC V 使用快速指南"中操作步骤,可以很便捷完成新建 "rgb\_led"工程。

(1) 新建 "rgb\_led" 工程

首先通过点击 start\_cmd.bat 文件来运行命令行,并进入到路径 "sdk\_env\_v0.10.0\hpm\_sdk\samples\rgb\_led",然后再命令行窗口执行命令 "generate\_project -b hpm6750evkmini",就会生成 SES 工程。



进入 "rgb led\hpm6750evkmini build\segger embedded studio",

然后,双击 "rgb\_led.emProject"文件,可以打开 "rgb\_led"的 SES 工程。 使用 Segger Embedded Studio 打开 rgb\_led 工程即可进行编译。





通过 "Project>Build reb\_led" 或者用快捷键 F7 来编译工程。





<u>File Edit View Sear</u>	ch <u>N</u> avigate	Project Bu	ld <u>D</u> ebug l <u>a</u> rget	lools <u>V</u>	<u>N</u> indow	Help	
Project Explorer	🖸 🗙	rgb_led. 👪	Build rgb_led		F7		
C Debug	•	$\leftarrow \rightarrow$	<u>Rebuild rgb_led</u>		Alt+F/		
Project Items	Code Da		a.t		CI.10. E1	•	
Solution 'rgb_led'			Build Solution		Shift+F		
Project 'rgb_led'	33.0K		Rebuild Solution		Alt+Shi	t+F7	
🔺 🖼 app 🛛 file	[3.4K]		Clean Solution				
B rgb_led.c	3.4K	82a	Compile rgb_led.c		Ctrl+F7		
boards 2 files			Duild and Dahun		CHLIT	- 6	
Components 1 file		-	Build and Debug		Cul+1, I		
drivers 31 files		1 1 1	Build and Run		Ctrl+1, (	trl+F5	
Soc 5 files		e la	Cance <u>l</u> Build		Ctrl+.		
Itoolchains 3 files			Ruild Configurations				
utils 2 files		10°	Sat Active Build Config	wration			
🖻 😺 Output Files			Set Active Build Config	guration		į.	
			Batch Build				
			Parallel and Unity Build	1		•	ED_PWM_CMP_INITIA
			Sho <u>w</u> Build Log		Ctrl+B, 0	Ctrl+T	CLOCK_NAME
		20	se				
		21	ror BOARD_RED_PW	M needs	to be o	defined	
		22	ULT				
		24	def BOARD_GREEN_	PWM			
		25	fine GREEN_PWM_I	RQ BOARD	_GREEN_	PWM_IF	ŧQ
		26	fine GREEN_PWM B	OARD_GRE	EN_PWM		
		Output					
				1			
		Show: Trans	cript 🔹 🍾	Output	•		
		1> Compili	ng 'hpm_debug_con	nsole.c'			
		6> Compili	ng 'pinmux.c'				
		2> Compili	ng 'board.C'				
		8) Compili	ng 'hom ffssi c'				

(3) 下载调试 "rgb\_led" 工程

使用 Segger Embedded Studio 进行 rgb\_led 调试。通过 "Debug>Go" 或者用快捷键 F5 来调试工程。

Project Evolorer		rab led c		Go	F5	
Ele Edit View Sear Project Explore	ch Navigate	Project         But           [gb_led.c]         1           2         3           3         4           5         6           7         7           9         10           11         12           13         14           14         15	ilid De Cc ← SF Cl ⊊I Cl ⊊I Cl ⊊I Cl ⊊I Cl ⊊I Fi Fi Fi	bug Targe Tools Go Break Stop Restart Toggle Breakpoint Breakpoints Step Juto Step Out Run To Cursor Auto Step Show Next Statement Set Next Statement	Window         Help           F5         Ctrl+.           Shift+F5         Ctrl+.           Ctrl+.Shift+F5         F0           F1         F10           Shift+F11         Ctrl+.F10           Alt+*         Shift+F1           Ctrl+.F10         Ctrl+.F10	
		17 18 19 20 21 22	fi fi 👓 se 🔍 rc di	Switch Debug Mode Quick Watch Debug With Ozone Options	Ctrl+F11 Shift+F9 Alt+F5	D_PWM_CMP_INITIAL_ZER
		23 24 25 26 Output Show: Tran 1> Compil 6> Compil 5> Compil	def B fine fine script ing 'pi ing 'bc	OARD_GREEN_PWM GREEN_PWM_IRQ BOAR GREEN_PWM BOARD_GR 	D_GREEN_PWM_IR EEN_PWM	Q 

(4) 调试 "rgb\_led" 工程: 断点使用。

在相应的文件编辑器窗口,通过在相应行点击鼠标右键,选中菜单项 "Toggle Breakpoint"或者使用 F9,可以打上断点。



		10		先楫半导体
🛇 rgb_led - SEGGEI	R Embedded	Studio for F	ISC-V V6.20a (64-bit) - Non-Commercial License (Stopped)	
<u>Eile Edit V</u> iew	<u>S</u> earch <u>N</u>	avigate <u>P</u> re	ject <u>B</u> uild <u>D</u> ebug T <u>a</u> rget <u>T</u> ools <u>W</u> indow <u>H</u> elp	
Project Explorer		🖂 ×	rgb_led.c	🛛 ×
C Debug		- 🖸 🧰	□ rgb_led    • int main()    • ♠    • ■    • ■    • •	Ç∃ç∃+∃ oo <sub>e</sub> ∃@
Project Items	Code	Data+RO	178 }	
Solution 'rgb_led'			179 pwm_update_raw_cmp_edge_aligned(pwm, pwm_cmp, duty);	
- Project 'rgb_lee	в 33.0К	36.0K	180 do_update = Taise;	
🔺 🖼 app 🛛 file	[3.4K]	[145]	<pre>182 turn_off_rgb_led(current);</pre>	
> b rgb_led.c	3.4K	145	<pre>183 pwm_disable_irq(pwm, PWM_IRQ_HALF_RELOAD);</pre>	
boards 2 file	s		184	
Components			⇒ 186 □ int main(void){	
drivers 31 file	55		187 uint32_t freq;	
soc 5 files	6		188 uintsz_t nw_event_cmp;	
> a tooicnains a	n		Toggle Breakpoint F9	
<ul> <li>Output Files</li> </ul>			Clear All Breakpoints     Ctrl+Shift+F9	
· · · · · · · · · · · · · · · · · · ·			Ioggle Bookmark Ctrl+F2	
			Glear Bookmarks In Document Ctrl+K, F2 PWM_CLOCK_NAME);	+
			* PWM_PERIOD_IN_MS - 1;	
			Show Eold Gutter	
			Show Disgnostic Icone In Gutter	
			= RED_PWM_CMP_INITIAL_ZERO;	
				+
			Ou 🗸 Enable Popup Diagnostics 🛛 🔯 🛃 🗙 Call Stack	E ×
			Find Error After Building	**
			Show, raiger 🔶 🔶 Output	<b>~</b>
			Function	Call Address
			int main()     int main()	0x01000F2C
			void reset_handler()	0.001005900

void reset\_handler() start() 0x01000050 L Erasing 36.5 KB of addresses 01000000 - 0100923b Erase successful Downloading 'rgb led.elf' to GDB Server 调试 "rgb\_led" 工程: 观察内存。 (5)

在相应的变量上点击右键,选中菜单项 "Add hw event cmp to Watch"

đ X

X

in x

或者使用快捷键 "Ctrl+T, Ctrl+W",可以将相应的变量增加到内存观察窗口。

o for RISC-V V6.20a (64-bit) - Non-Commercial License (Stopped) Project Build Debug Target Tools Window Help

(	rgb_led.c						×	Watch 1	🏹 🐺 📑 🌄 🗙
	□ rgb_led	•	• int main()	-★★ ▶ ■ ← 5 🗉	Ç⊒ ç⊒ +⊒	00 4	1	Never - X <sub>2</sub> X <sub>8</sub>	X <sub>10</sub> X <sub>16</sub> X <sup>±</sup> <sub>10</sub> 'X' 🐺 🎇 盾 🖹
	187		uint32_t frea:					Expression	Value
	188		uint32_t nw_event_cr	Go To Definition	Alt+G			<pre>&gt; leds</pre>	<array></array>
	189		board init():	Co To Dedenation	Altechilter			hw_event_cmp	0×0000008
	190		board init rgh nwm r	GO TO Declaration	All+Shill+G				
	• 192		printf("rgb led exam	Go to Included File					
	193			Go to Eunction		•			
	• 194		freq = clock_get_fre #	ri-d p.(	Alt . D				
	<ul> <li>195</li> </ul>		leds[red].reload = f	Find Keterences	Alt+K				
	▶ 196		leds[red].pwm = AED_	Find Extras		•			
	<ul> <li>197</li> </ul>		leds[red].pwm_ch F	Toggle Preskpoint	EQ				
	<ul> <li>198</li> </ul>		<pre>leds[red].pwm_cmp = 0</pre>		15				
	• 199		leds[red].pwm_cmp_r	Edit Breakpoint					
	• 200		leds[red].pwm_irq =	<u>R</u> un To Cursor	Ctrl+F10				
	201		from - clock got fre	Set Next Statement	Shift+F10		/		
	202		leds[green] reload -	Show Next Statement	A				
	<ul> <li>203</li> <li>204</li> </ul>		leds[green].pwm = GF	- Silow Wext Statement	AILT			Registers 1	×
	• 205		leds[green].pwm ch = ?	Quick Watch "hw_event_cmp"	Shift+F9	1			
	<ul> <li>206</li> </ul>		leds[green].pwm cmp 🎭	Add "hw event cmp" to Watch	Ctrl+T. Ctrl+'	w		🕑 Groups 🛛 🦃 🏈 🔿	🕑 🔠 😽 🖼 🖬 🖬 Search
	<ul> <li>207</li> </ul>		leds[green].pwm_cmp_	Set Breakpoint on "hw event cmp"				Name Value	
	► 208		leds[green].pwm_irq	set breakpoint on nw_event_cmp				Valac	
			*	Locate Memory	Ctrl+T, Y	1	+		

还能通过内存窗口控件标签,来切换观察变量的类型。可以查看局部变量、

#### 全局变量、自定义内存等。

RISC-V V6.20a (64-bit) - Non-Commercial License (Stopped) - 0 oject <u>B</u>uild <u>D</u>ebug T<u>a</u>rget <u>T</u>ools <u>W</u>indow <u>H</u>elp rgb\_led.c 🗛 🐺 🐺 🗙 Globals • ♠ ♥ | ▶ ■ ← ♥∃ [∃ ɕヨ +⊒ ∞ ݷ≣ 🦳 □ rgb\_led X<sub>2</sub> X<sub>8</sub> X<sub>10</sub> X<sub>16</sub> X<sup>±</sup><sub>10</sub> 'X' ↓ 1 🗈 🛱 • int main() 1 uint32\_t freq; uint32\_t hw\_event\_cmp; Expression Value 188 red (0) false current 188 189 190 191 192 do\_update \* \* \* board\_init(); leds <array> board\_init\_rgb\_pwm\_pins();
printf("rgb led example\n"); stdout 0x01080038 193 194 195 196 197 198 199 200 201 202 203 204 205 206 207 208 freq = clock\_get\_frequency(RED\_PWM\_CLOCK\_NAME); leds[red].reload = freq / 1000 \* PMM\_ERIOD\_IN\_MS - 1; leds[red].pwm = RED\_PWM; leds[red].pwm\_cmp = RED\_PWM\_OUT; leds[red].pwm\_cmp\_eRED\_PWM\_CMP; leds[red].pwm\_cmp\_initial\_zero = RED\_PWM\_CMP\_INITIAL\_ZERO; leds[red].pwm\_irq = RED\_PWM\_IRQ; \* \* \* \* \* \* \* freq = clock\_get\_frequency(GREEN\_PWM\_CLOCK\_NAME); leds[green].reload = freq / 1000 \* PWM\_PERIOD\_IN\_MS - 1; leds[green].pwm\_ch = GREEN\_PWM\_OUT; leds[green].pwm\_cmp = GREEN\_PWM\_CMF; leds[green].pwm\_cmp\_initial\_zero = GREEN\_PWM\_CMP\_INITIAL\_ZERO; leds[green].pwm\_ ing = GREEN\_PWM\_IRO: ...... Registers 1  $\odot$  Groups  $\Rightarrow$   $\Rightarrow$   $\Rightarrow$ Name Value 调试 "rgb\_led" 工程: 监测寄存器。 (6)



通过 "View>Registers" 来显示寄存器窗口, 进而观察程序调试过程中,

#### 各种寄存器中相应值的变化情况。



- (7) 调试 "rgb\_led" 工程: 函数调用栈。
  - 通过 "View>Call Stack" 或者快捷键 "Ctrl+Alt+S" 来显示函数调用栈窗

口,进而观察程序调试过程中,函数调用情况。

rgb\_led - SEGGER Embedded Studio for RISC-V V6.20a (64-bit) - Non-Commercial License (Stopped)



更详细的 SES 使用方法,可以参考 SES 的帮助手册。



## 8 HPM SDK 常见问题

问题描述	解决方法							
Debug 无法连接 EVK, 无	EVK 上均设计有 BOOT 模式开关,用户需要保证开关位于正确的组合,才							
法烧写, Flash 内程序可以	能进行调试。							
运行。	Boot 模式选择在 SDK 中的 Doc 目录下的 EVK 使用说明书有详细的描述。							
SES 无法正确编译	请确认下载的 SES 的版本为 Risc-V 版本。							
SES 中无法输入中文,中	不建议在 SES 中使用中文注释。							
文乱码								
SES 中提示 GDB server	检查是否正确安装 FT2232 驱动,正确安装后,设备管理器中应存在 1 个							
出错	串口和一个 USB 设备如下:							
	> <b>邊</b> 电池							
	→ 量 端口 (COM 和 LPT) 量 USB Serial Port (COM6)							
	> 圖 固件							
	> 👤 计算机							
	> 🚺 监才 > 🛄 内存技术设备							
	> 同人机接口设备							
	> ■ 软件设备 > ■ 软件组件							
	→ 「」 声音、视频和游戏控制器 ● ほしていせいせいというを							
	<ul> <li>♥ 通用串行总线设备</li> <li>● p. 155333 45</li> </ul>							
	PUGI RS232-HS > ■ 网络适配器							
基于 windows 的操作系	1、 在 Windows 的任务管理器找到挂死的 SES 进程。							
统,在SES中通过register	n Task Manager							
窗口访问未开 clock 模块	File Options View							
寄存器, 会出现 SES 挂死。	Processes Performance App history Startup Users Details Services							
现象:	Name Status CPU /							
1, 用户打开了寄存器窗								
口,	Apps (16)							
2, 在 download 加	> G Foxit PDF Reader 11.2 (32 bit) 0%							
debug 过程中,寄存	> C Microsoft Edge (23)         0%         4							
器窗口里有需要初始	Mobaxterm (32 bit) (2)     U%							
化 clk 的 soc 寄存器。	SEGGER Embedded Studio for Ri     13,9%							
	i openocd.exe							
	SEGGER Embedded Studio for Not responding 0% 1							
	> 🐼 SEGGER Embedded Studio for RI 0%							
	> 🚺 Snip & Sketch (2) 0%							
	2、 从挂死的 SES 进程中找到关联的 openocd 进程,并将其结束运							
	行。							

Tel Manager				10 07	
Task Manager					
File Options View					
Processes Performance App	history Startup Users Deta				
~		30%	54%	0%	0%
Name	Status	CPU	Memory	Disk	Network
Apps (16)					
> G Foxit PDF Reader 11.2	(32 bit)	0%	5.9 MB	0 MB/s	0 Mbps
> 💽 Microsoft Edge (23)		0%	439.1 MB	0 MB/s	0 Mbps
> 💐 MobaXterm (32 bit) (2)	)	0%	2.5 MB	0 MB/s	0 Mbps
👻 🧇 SEGGER Embedded Stu	udio for RI	18.0%	128.4 MB	0 MB/s	0 Mbps
index.exe		0%	4.1 MB	0 MB/s	0 Mbps
i openoc		18.0%	3.5 MB	0 MB/s	0 Mbps
SEGGER End tas	Donding	0%	120.9 MB	0 MB/s	0 Mbps
> SEGGER I Provid	e feedback	0%	4.6 MB	0 MB/s	0 Mbps
3	板				
4、   更改 SES	中 GDB server 设置	重,将"Res	set and S	stop Co	mmanc
的选项改	为"reset init",保 <sup>;</sup>	存设置后,	SES 可以	以重新调	试
Search Options	2				Show Modified
La Code	10				
Assembler		Vellue			
Assembler Build & Code Analyzer	GDB Server Host	localhost			
Attembler Uption build Code Analyter Code Consolition Compiler	CDB Server Host Type GDB Server Command Line	Value localhost Custon (inots) D./repos/sdk_erw/tools/o	ipenacci/apenacd.exe =1 50	HPM_SDK_BASE/boards/	openotd/probes/ft2232
Accenden: Uppe Dudi Code Analyter Code Canadition Complet External build File	GDB Server Host Type GDB Server Command Line Auto State Colo Server Pert Reset and Stop Command	Value locelhost Custon linets D/reposited, envitools// Ves latents 2,222 Dents rest Init	ipenood/openotd.exe + 50	HPM_SDK_BASEI/boards/	openotal/probes/ft2232
Azembler Winner Buld Cole Analyre Cole Conston Campor Campor File File Libraries Libraries Libraries	CORE Server Host Type GOD Server Command Line Juno Start CORE Server Pert Restand Step Command Genore Constant Errors Jakow Minning Arctis During Execution	Volue localhost Custon Innets Directorised, emitools/r Visi Minets 2000 Custon reset mit No module my module	openood/openood.exe + 50	HPM_SDK_BASE//boards/	openocd/probes/ft2232
Atembler Winker Build Cold-Analyre Code Conston Compiler Edimine Build Ubraries Ubraries Ubraries Ubraries Ubraries Ubraries	CODE Server Hest Type server Contended Line: Judo Stand CoDE Service Part Baret and Stape Contended Baret and Stape Contended Baret and Stape Contended Register Access Register Access Register Access Register Access Baret and Stape Contended Register Access Baret and Stape Contended Register Access Baret and Stape Contended Register Access Baret and Stape Contended Register Access Baret and Stape Contended Baret and Stape Conten	Value localhost Custon Shoels Drivenosyske revitosis/ versional reset init nervol set General and individual Software Only mediad	openocd/openocd.exe + 50 mostec	HPM_SDK_BASE/boards/	openoca/probes/ft2232
Attembler Windows Build Code Analyze Code Centration Compiler Defamine Build Prison Build Uniter Preprocessor Preprocessor Preprocessor Function	CODE Server Heat Thest Systems Connected line Systems Connected line Systems Connected Part Rest and Stop Connected Rest and Stop Connected Rest and Stop Connected Rest and Rest Rest And Rest And Rest Rest And Rest And Rest Rest And Rest And Rest Rest And Rest And Res	Value localhost Custon finetis Origination (include origination) Account (include origination) No mostle origination Contrained (individual Software Only medical None Software Software)	openocd/openocd.exe + 50 montfact	HPM_SOK_BASE/boards/	openo cd/probes/ft/2232
Attembler Works	COB Server Command Line Auto: Start Command Line Auto: Start Command Line Auto: Start Command Line Execution Store Resident Access Resident Access Compart Exection Compart Execution Resident Threes Connect Timesd Write Timesd	Value localhost Custon inners Directoris (di emitodia) 2022 dente rest nice Nor market Nor market Nor market Rest nice Central and Individual Schwarc Unity maillial Norte S seconds S seconds S diversit S seconds	openood/openoid.exe + 50 monfeel	HPM_SDK_BASE/boards/	openocd/probes/tt2232
Accender: University of a Cost Analyser Cost Analyser Cost Analyser Cost Cost An	COB Server Test Visit Command Line Auto Start Command Line Auto Start CoB Server Test Common Server Start Common Server Start Common Server Start Common Server Start Common Server Start Common Server Start Start Start Server Start Start Start Server Start Start Server Start Start Server Start Start Server Start Start Server Start Sta	Value location Custon Frent D. (reported entropy) Register Registe	openood/openoid.exe + \$9	HPM_SDK_BASEI/boards/	openo:d/probes/RE2532
Accembler: University of a Code Analyser Code Analyser Code Analyser Code Code Analyser Code Code Code Code Code Code Code Code	COB Server Freit Troop Server Command Line Auto Start Command Line Auto Start Command Ignore Checksana Enors Alabor Mannay Access Registra Access Alabor Mannay Access Registra Access Taropt CMR Frei Registra Access Registra Access	Value location Custon Henris D. (reported environment Recent and the second Recent and Individual Centr	upenaci/apenacid.exe + 50	HPM_SDK_BASE//boards/	openo:d/probes/#223