



CV184X RT-Thread 编译使用手册

Version: 1.0

Release date: 2025-09-04

©2025 北京晶视智能科技有限公司
本文件所含信息归北京晶视智能科技有限公司所有。
未经授权，严禁全部或部分复制或披露该等信息。

目录

1	声明	2
2	编译环境	3
2.1	安装 python 及 pip	3
2.2	安装 SCons 工具	3
2.3	SDK 文件说明	4
2.4	如何编译 RT-Thread	6
2.5	镜像说明	8
3	运行说明	9
4	RT-Thread 指令集使用	11
4.1	系统信息	11
4.1.1	help	11
4.1.2	reboot	12
4.1.3	clear	12
4.2	内存操作命令	12
4.2.1	devmem	12
4.2.2	dumpmem	12
4.2.3	free	12
4.3	进程信息	13
4.3.1	进程整体信息	13
4.3.1.1	ps	13
4.3.1.2	list	13
4.3.1.3	backtrace	13
4.4	文件系统操作命令	13
4.4.1	ls	13
4.4.2	cd	13
4.4.3	pwd	13
4.4.4	cat	13
4.4.5	mkdir	14
4.4.6	rm	14
4.4.7	cp	14
4.4.8	mv	14
4.4.9	echo	14
4.4.10	tail	14
4.4.11	df	14
4.4.12	mount	14
4.4.13	umount	15
4.4.14	mkfs	15
4.5	硬件操作命令	15

4.5.1	pin	15
4.5.2	single_core_cache	15
4.6	添加自定义命令	15

修订记录

Revision	Date	Description
v1.0	2025/09/04	Initial

1 声明



法律声明

本数据手册包含北京晶视智能科技有限公司（下称“晶视智能”）的保密信息。未经授权，禁止使用或披露本数据手册中包含的信息。如您未经授权披露全部或部分保密信息，导致晶视智能遭受任何损失或损害，您应对因之产生的损失/损害承担责任。

本文件内信息如有更改，恕不另行通知。晶视智能不对使用或依赖本文件所含信息承担任何责任。本数据手册和本文件所含的所有信息均按“原样”提供，无任何明示、暗示、法定或其他形式的保证。晶视智能特别声明未做任何适销性、非侵权性和特定用途适用性的默示保证，亦对本数据手册所使用、包含或提供的任何第三方的软件不提供任何保证；用户同意仅向该第三方寻求与此相关的任何保证索赔。此外，晶视智能亦不对任何其根据用户规格或符合特定标准或公开讨论而制作的可交付成果承担责任。

联系我们

地址 北京市海淀区丰豪东路 9 号院中关村集成电路设计园（ICPARK）1 号楼

深圳市宝安区福海街道展城社区会展湾云岸广场 T10 栋

电话 +86-10-57590723 +86-10-57590724

邮编 100094（北京）518100（深圳）

官方网站 <https://www.sophgo.com/>

技术论坛 <https://developer.sophgo.com/forum/index.html>

2 编译环境

2.1 安装 python 及 pip

1. 安装 Python3
2. 下载 PIP 脚本
3. 安装 pip, 执行如下命令

```
python get-pip.py
```

4. 安装 python 的 serial 和 usb 驱动模块, 执行如下命令

```
pip3 install pyserial pyusb
```

5. 安装 yaml 模块, 执行如下命令

```
pip3 install pyyaml
```

注解: 如果上面 pyyaml 安装失败, 可以使用下面命令安装

```
pip3 install pyyaml -i http://pypi.douban.com/simple --trusted-host pypi.douban.com
```

注意: PS: ubuntu 版本推荐高于 20.0.4

2.2 安装 SCons 工具

1. Linux 环境中使用指令安装 scons

```
sudo apt-get install scons
```

2. 使用 scons -V 查看对应版本, 确保版本处于或者高于 3.1.2

```
jingjing.tang@cvitek:board (HEAD)$ scons -v
SCons by Steven Knight et al.:
  script: v3.1.2.bee7caf9defd6e108fc2998a2520ddb36a967691, 2019-12-17 02:07:09, by bdeegan on octodog
  engine: v3.1.2.bee7caf9defd6e108fc2998a2520ddb36a967691, 2019-12-17 02:07:09, by bdeegan on octodog
  engine path: ['/usr/local/lib/python3.8/dist-packages/scons/SCons']
Copyright (c) 2001 - 2019 The SCons Foundation
```

2.3 SDK 文件说明

SDK 主要的文件夹和目录如下：

```
.
├── bsp                                ## 板级支持包
├── cvitek
│   ├── board_env.sh
│   ├── c906_little
│   │   ├── applications
│   │   ├── audio
│   │   ├── board
│   │   ├── comm
│   │   ├── emptybin
│   │   ├── Kconfig
│   │   ├── rgn
│   │   ├── rtconfig.h
│   │   ├── rtconfig.py
│   │   ├── rtos_types.h
│   │   ├── SConscript
│   │   └── SConstruct
│   ├── combine-fip.sh
│   ├── cv18xx_aarch64
│   ├── cv18xx_boot
│   ├── cv18xx_risc-v
│   ├── cv18xx_risc-v-m-mode
│   ├── cvi_boards
│   ├── cvi_chips
│   ├── cvi_common
│   ├── docs
│   ├── mkimage
│   ├── mksdimg.sh
│   ├── README.md
│   └── README_milkduo.md
├── ChangeLog.md
├── components
├── cvi_comps
│   ├── cvi_json_c
│   ├── cvi_mpi_gdc
│   ├── cvi_mw_audio
│   ├── cvi_mw_bin
│   ├── cvi_mw_isp
│   ├── cvi_mw_isp_ae
│   ├── cvi_mw_isp_af
│   ├── cvi_mw_isp_algo
│   ├── cvi_mw_isp_awb
│   ├── cvi_mw_isp_bin
│   └── cvi_mw_isp_common
```

(下页继续)

(续上页)

```

├── cvi_mw_isp_daemon
├── cvi_mw_ive
├── cvi_mw_mipi_tx
├── cvi_mw_rgn
├── cvi_mw_sys
├── cvi_mw_vdec
├── cvi_mw_venc
├── cvi_mw_vi
├── cvi_mw_vpss
├── include
├── SConscript
├── documentation
├── examples
├── include
├── Kconfig
├── libcpu
├── LICENSE
├── README_de.md
├── README_es.md
├── README.md
├── README_zh.md
├── src
└── tools

```

1. 主要目录说明:

bsp/cvitek/: CVITEK 芯片系列 BSP 的根目录, 所有相关代码都集中在这里

- bsp/cvitek/c906_little: 小核系统配置目录。
- bsp/cvitek/cv18xx_boot/: 芯片的启动引导相关。
- cvi_common: 公共代码库, 包括硬件抽象层 (HAL)、芯片外设驱动 (drivers)、工具代码、测试代码等。
- mkimage、mkxsdimg.sh: 镜像文件制作。

components/cvi_comps/cvi_mpi: CVITEK 多媒体中间件 (Middleware) 组件库。这些组件为应用提供了访问芯片硬件加速功能的 API。

- cvi_mw_audio/: 音频中间件, 提供音频采集 (AI)、播放 (AO)、编码 (AENC)、解码 (ADEC) 等功能。
- cvi_mw_isp/ 及其子目录: 图像信号处理中间件, 包含 AE(自动曝光)、AF(自动对焦)、AWB(自动白平衡) 等算法和守护进程。
- cvi_mw_vi/: 视频输入 (Video Input) 中间件, 用于摄像头传感器采集。
- cvi_mw_venc/: 视频编码 (H.264/H.265/JPEG) 中间件。
- cvi_mw_vdec/: 视频解码中间件。
- cvi_mw_vpss/: 视频处理子系统 (Video Process Sub-System) 中间件, 用于缩放、裁剪、像素格式转换等。
- cvi_mw_rgn/: 区域覆盖 (Region) 中间件, 用于在视频画面上叠加 OSD、遮挡块等。
- cvi_mw_mipi_tx/: MIPI DSI/CSI 传输中间件。

- `cvi_mw_ive/`: 智能视频分析 (Intelligent Video Engine) 中间件, 用于移动侦测、遮挡检测等。
- `cvi_mw_sys/`: 系统控制中间件, 提供系统初始化、内存分配 (VB 池) 等基础服务。
- `cvi_mw_bin/`: 可能存放一些预编译的算法库或固件。
- `cvi_json_c/`: JSON 解析库。
- `include/`: 上述所有中间件组件的公共头文件目录。

2. Scons 工具使用说明

<https://github.com/sophgo/rt-thread/blob/cv184x-v6.x/documentation/scons/scons.md>

SConstruct 和 SConscript 主要用于定义和开启功能宏比如选择不同的 Sensor, 以及需要依赖的组件选择

2.4 如何编译 RT-Thread

- 设定环境变量 (以 `cv1842hp_wevb_0014a_spinor` 为例)

```
$ source build/envsetup_soc.sh
```

Usage:

(1) `menuconfig` - Use menu to configure your board.

ex: `$ menuconfig`

(2) `defconfig $CHIP_ARCH` - List EVB boards(\$BOARD) by `CHIP_ARCH`.

`** cv184x ** -> ['cv184x', 'cv1841c', 'cv1842cp', 'cv1842hp', 'cv1843hp']`

ex: `$ defconfig cv184x`

(3) `defconfig $BOARD` - Choose EVB board settings.

ex: `$ defconfig cv1842hp_wevb_0014a_spinor`

ex: `$ defconfig cv1842cp_wevb_0015a_spinand`

- 选定 EVB `cv1842hp_wevb_0014a_spinor`

```
$ defconfig cv1842hp_wevb_0014a_spinor
```

```
===== Environment Variables =====
```

```
PROJECT: cv1842hp_wevb_0014a_spinor, DDR_CFG=ddr3_2133_x16
```

```
CHIP_ARCH: CV184X, DEBUG=0
```

```
SDK VERSION: musl_arm, RPC=0
```

```
ATF options: ATF_KEY_SEL=default, BL32=1
```

```
Linux source folder:linux_5.10, Uboot source folder: u-boot-2021.10
```

```
ENABLE_DUAL_OS: y
```

```
CROSS_COMPILE_PREFIX: arm-none-linux-musleabihf-
```

```
ENABLE_BOOTLOGO: 0
```

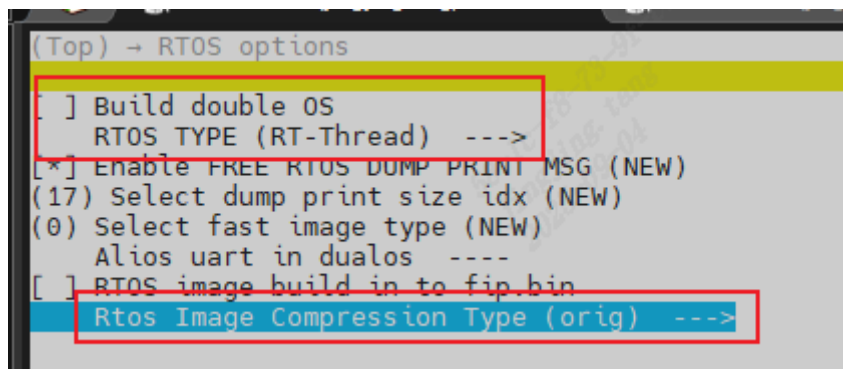
```
Flash layout xml: build/boards/cv184x/cv1842hp_wevb_0014a_spinor/partition/partition_spinor.xml
```

```
Sensor tuning bin: cvsens_cv2003
```

```
Output path: install/soc_cv1842hp_wevb_0014a_spinor
```

- 单系统配置

通过 menuconfig 命令，勾选对应配置，即可编译 RT-Thread 镜像。



· 编译 RT-Thread

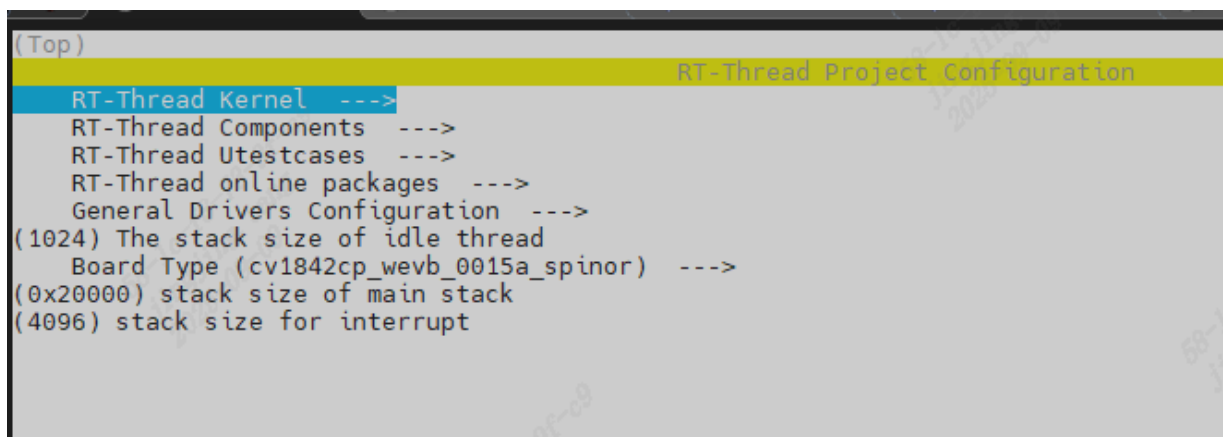
RT-Thread 镜像文件被打包到启动镜像中通常通过一条命令。

```
$ build_uboot
Run build_uboot() function
Run _link_uboot_logo() function
.....
[TARGET] rtt-build
.....
INFO: raw2cimg small part size
INFO: size:b5c42, offset:200000, part_sz:100000, crc:38896f66
INFO: Packing yoc.bin done!
```

编译完成后，生成的 RT-Thread 镜像文件 yoc.bin 将被拷贝到 install 目录中。

· 配置 RT-Thread

通过 menuconfig_rtt 命令，在图像化配置界面中配置 RT-Thread。



图像化界面的操作请参考《CV184X SDK 编译使用手册》中与编译组态设定相关章节

2.5 镜像说明

在 `bsp/cvitek/c906_little/` 的生成文件：

1. `rtthread.elf`: 完整的可执行文件等，可以被 `objdump` 或 `readelf` 工具解析
2. `rtthread.bin`: 运行程序 `bin` 档案，SDK 编译过程会拷贝为 `install` 下的 `yoc.bin` 文件，是烧录到 Flash 的小核镜像文件
3. `rtthread.map`: 内存映射文件

`components/cvi_comps/cvi_mpi/*/*.a`: 静态库

3 运行说明

1. 烧录完毕后关闭电源等待 1-2S 钟，存在电容放电需要等释放完毕
2. 上电有如下打印证明内核启动完毕

```
RT_HW_HEAP_BEGIN:800fe3c0 RT_HW_HEAP_END:801a0000 size: 662592

\ | /
- RT -   Thread Operating System
/ | \   5.2.0 build Sep  3 2025 15:56:50
2006 - 2024 Copyright by RT-Thread team
Hello, RISC-V1!
dump_print_enable & log will not print
[cvi_spinlock_init] success
prvCmdQuRunTask run
Enter cvitek audio ssp algorithm test mode
audio_ssp first trigger test begin.....!!!
audio_ssp first trigger test end!
msh />
```

3. 输入串口回车可以正常输入输出，并且有 msh 打印即运行正常

```
msh />help
RT-Thread shell commands:
dumpmem      - "dumpmem < addr > < size >"
devmem       - "devmem < addr >"
reboot       - reset machine
single_core_cache - Single core cache operation test
pin          - pin [option]
list         - list objects
version      - show RT-Thread version information
clear        - clear the terminal screen
free         - Show the memory usage in the system
ps           - List threads in the system
help         - RT-Thread shell help
tail         - print the last N - lines data of the given file
echo         - echo string to file
df           - disk free
umount       - Unmount the mountpoint
mount        - mount <device> <mountpoint> <fstype>
mkfs         - format disk with file system
mkdir        - Create the DIRECTORY.
pwd          - Print the name of the current working directory.
cd           - Change the shell working directory.
rm           - Remove(unlink) the FILE(s).
cat          - Concatenate FILE(s)
mv           - Rename SOURCE to DEST.
cp           - Copy SOURCE to DEST.
ls           - List information about the FILES.
backtrace    - print backtrace of a thread
msh />
```

4. 小核 pc 值查看, 可以观察 pc 值在变化:devmem 0x1901070

4 RT-Thread 指令集使用

4.1 系统信息

4.1.1 help

该命令将打印 RT-Thread 所有命令列表并简要描述每个命令的作用。

```
msh />help
RT-Thread shell commands:
dumpmem          - "dumpmem < addr > < size >"
devmem           - "devmem < addr >"
reboot           - reset machine
single_core_cach - Single core cache operation test
pin              - pin [option]
list             - list objects
version          - show RT-Thread version information
clear            - clear the terminal screen
free             - Show the memory usage in the system
ps               - List threads in the system
help             - RT-Thread shell help
tail             - print the last N - lines data of the given file
echo             - echo string to file
df               - disk free
umount           - Unmount the mountpoint
mount            - mount <device> <mountpoint> <fstype>
mkfs             - format disk with file system
mkdir            - Create the DIRECTORY.
pwd              - Print the name of the current working directory.
cd               - Change the shell working directory.
rm               - Remove(unlink)the FILE(s).
cat              - Concatenate FILE(s)
mv               - Rename SOURCE to DEST.
cp               - COPY SOURCE to DEST.
ls               - List information about the FILEs.
backtrace        - print backtrace of a thread
```

4.1.2 reboot

重启 RT-Thread 系统，Linux 系统也会随之重启。

4.1.3 clear

清除终端屏幕内容。

4.2 内存操作命令

4.2.1 devmem

直接读写物理内存地址。使用时需格外谨慎。

```
msh />devmem <address> [value]
```

参数说明：

address：要操作的物理地址

value：（可选）要写入的值，如果省略则读取该地址的值

4.2.2 dumpmem

以十六进制格式显示指定内存区域的内容。

```
msh />dumpmem <start-address> <size>
```

参数说明：

start-address：起始内存地址

size：要显示的内存大小（字节数）

4.2.3 free

显示系统内存使用情况，包括总内存、已用内存和剩余内存。

4.3 进程信息

4.3.1 进程整体信息

4.3.1.1 ps

列出系统中所有线程的状态信息。

4.3.1.2 list

列出系统内核对象（如信号量、互斥量、消息队列等）。

4.3.1.3 backtrace

显示指定线程的调用栈回溯信息，用于调试。

4.4 文件系统操作命令

4.4.1 ls

列出目录内容。

4.4.2 cd

改变当前工作目录。

4.4.3 pwd

显示当前工作目录的路径。

4.4.4 cat

显示文件内容。

4.4.5 mkdir

创建新目录。

4.4.6 rm

删除文件或目录。

4.4.7 cp

复制文件或目录。

4.4.8 mv

移动或重命名文件/目录。

4.4.9 echo

向文件写入字符串内容。

4.4.10 tail

显示文件的最后几行内容。

4.4.11 df

显示文件系统的磁盘使用情况。

4.4.12 mount

挂载文件系统。

4.4.13 umount

卸载已挂载的文件系统。

4.4.14 mkfs

在块设备上创建文件系统。

4.5 硬件操作命令

4.5.1 pin

GPIO 引脚操作工具，用于读取或设置引脚状态。

pin [options]

选项：

不带参数：显示所有引脚状态

pin <pin> <value>：设置指定引脚的值（0 或 1）

pin <pin>：读取指定引脚的值

4.5.2 single_core_cache

单核缓存操作测试命令，用于调试和测试缓存功能。

4.6 添加自定义命令

在 RT-Thread 操作系统中，MSH_CMD_EXPORT 是一个非常重要的宏，它用于将自定义的函数导出为 FinSH 命令行组件（通常称为 MSH，即 Module Shell）的命令。

在 CV184X SDK 中，用 MSH_CMD_EXPORT 宏和 Kconfig 配置，定义了一些 RT-Thread 上的测试命令，以 gpio test 为例：

1. 配置

通过 menuconfig_rtt 命令勾选，如下图

```
(Top) → General Drivers Configuration
RT-Thread Project Configuration
rootfs type (ROMFS) --->
[*] Using DesignWare DMA
[*] Using UART --->
[ ] Using I2C ----
[ ] Using ADC ----
[ ] Using SPI
[ ] Enable Watchdog Timer ----
[ ] Enable TIMER ----
[ ] Using PWM ----
[ ] Enable Secure Digital Host Controller
[ ] Enable TPU
[ ] ENABLE DRIVER TOOLS ----
[*] ENABLE_SELFTEST --->
[*] Enable Mailbox
```

```
(Top) → General Drivers Configuration → ENABLE_SELFTEST
RT-Thread Project Configuration
[*] Enable GPIO_SELFTEST
[ ] Enable SPI_SELFTEST
[ ] Enable TPU_SELFTEST
[ ] Enable BSP_TEST_UART
[ ] Enable BSP_TEST_SD
[ ] Enable BSP_TEST_WDT
[ ] Enable BSP_TEST_TIMER
[ ] Enable BSP_TEST_ADC
[ ] Enable BSP_TEST_PWM
```

2. 命令行执行

```
RT-Thread shell commands:
gpio_test - GPIO driver test
reboot - reset machine
single_core_cache - Single core cache operation test
pin [option] - pin [option]
list - list objects
version - show RT-Thread version information
clear - clear the terminal screen
free - Show the memory usage in the system
ps - List threads in the system
help - RT-Thread shell help
tail - print the last N - lines data of the given file
echo - echo string to file
df - disk free
umount - Unmount the mountpoint
mount - mount <device> <mountpoint> <fstype>
mkfs - format disk with file system
mkdir - Create the DIRECTORY.
pwd - Print the name of the current working directory.
cd - Change the shell working directory.
rm - Remove(unlink) the FILE(s).
cat - Concatenate FILE(s)
mv - Rename SOURCE to DEST.
cp - Copy SOURCE to DEST.
ls - List information about the FILEs.
backtrace - print backtrace of a thread

msh />gpio
gpio_test
msh />gpio_test

|-- Starting GPIO Driver Test --
```

3. 测试命令的 MSH_CMD_EXPORT 宏定义脚本

```
$ cat rt-thread/bsp/cvitek/cvi_common/cvi_test/bsp/gpio_test.c
static int gpio_test(int argc, char **argv)
{
    rt_kprintf("\n--- Starting GPIO Driver Test ---\n");

    .....
}

MSH_CMD_EXPORT(gpio_test, GPIO driver test);
```