



## Hi3861V100 / Hi3861LV100 常见问题

### FAQ

文档版本 03

发布日期 2020-07-29

版权所有 © 上海海思技术有限公司2020。保留一切权利。

非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。

## 商标声明



**HISILICON**、海思和其他海思商标均为海思技术有限公司的商标。

本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

## 注意

您购买的产品、服务或特性等应受海思公司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的购买或使用范围之内。除非合同另有约定，海思公司对本文档内容不做任何明示或默示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

# 上海海思技术有限公司

地址：            深圳市龙岗区坂田华为总部办公楼    邮编：518129

网址：            <https://www.hisilicon.com/cn/>

客户服务邮箱：  [support@hisilicon.com](mailto:support@hisilicon.com)



## 前言

### 概述

本文档主要介绍Hi3861V100、Hi3861LV100解决方案中常见的问题处理和解决办法。

### 产品版本

与本文档相对应的产品版本如下。

产品名称	产品版本
Hi3861	V100
Hi3861L	V100




### 读者对象

本文档主要适用于以下工程师：



- 技术支持工程师
- 软件开发工程师

### 符号约定

在本文中可能出现下列标志，它们所代表的含义如下。

符号	说明
 <b>危险</b>	表示如不可避免则将会导致死亡或严重伤害的具有高等级风险的危害。
 <b>警告</b>	表示如不可避免则可能导致死亡或严重伤害的具有中等级风险的危害。
 <b>注意</b>	表示如不可避免则可能导致轻微或中度伤害的具有低等级风险的危害。



符号	说明
 须知	用于传递设备或环境安全警示信息。如不可避免则可能会导致设备损坏、数据丢失、设备性能降低或其它不可预知的结果。 “须知”不涉及人身伤害。
 说明	对正文中重点信息的补充说明。 “说明”不是安全警示信息，不涉及人身、设备及环境伤害信息。

## 修改记录

文档版本	发布日期	修改说明
03	2020-07-29	在“ <a href="#">3.2.2.3.3 CPU寄存器信息</a> ”中新增 <a href="#">表3-6</a> 。
02	2020-06-28	<ul style="list-style-type: none"><li>更新“<a href="#">3.1.2 看门狗超时时间问题</a>”的内容。</li><li>新增“<a href="#">3.1.4 业务任务频繁触发看门狗</a>”小节。</li></ul>
01	2020-04-30	第一次正式版本发布。 <ul style="list-style-type: none"><li>新增“<a href="#">3 系统异常类</a>”章节。</li></ul>
00B02	2020-02-12	<ul style="list-style-type: none"><li>更新“<a href="#">2.1.1 HiBurn工具与串口连接问题</a>”中<a href="#">图2-1</a>。</li><li>更新“<a href="#">2.1.2 HiBurn工具与串口连接后打印窗口一直打印“errno”</a>”的<a href="#">解决方法</a>。</li></ul>
00B01	2020-01-15	第一次临时版本发布。



# 目录

前言.....	i
<b>1 SDK 开发环境和 SDK 使用类.....</b>	<b>1</b>
1.1 SDK 开发环境问题.....	1
1.1.1 Python 库安装失败.....	1
1.1.1.1 Python 库版本与 Python 版本冲突.....	1
1.1.1.2 安装工具库时 Linux 系统相关工具缺失.....	2
1.1.2 SCons 版本错误.....	2
1.1.2.1 SCons 版本与 Python 版本冲突.....	3
1.1.2.2 构建脚本中对版本描述与实际版本冲突.....	3
1.1.3 依赖包问题.....	4
1.2 SDK 使用问题.....	4
1.2.1 Python 版本错误.....	4
1.2.2 未指定编译器.....	5
<b>2 工具类.....</b>	<b>6</b>
2.1 HiBurn 工具常见问题.....	6
2.1.1 HiBurn 工具与串口连接问题.....	6
2.1.2 HiBurn 工具与串口连接后打印窗口一直打印“errno”.....	8
<b>3 系统异常类.....</b>	<b>9</b>
3.1 看门狗常见问题.....	9
3.1.1 看门狗简介.....	9
3.1.2 看门狗超时时间问题.....	9
3.1.3 看门狗自定义超时时间失败问题.....	9
3.1.4 业务任务频繁触发看门狗.....	10
3.2 死机信息及死机问题.....	10
3.2.1 死机信息简介.....	10
3.2.2 死机问题定位.....	10
3.2.2.1 查看导出死机信息.....	10
3.2.2.2 死机信息定位步骤.....	10
3.2.2.3 死机信息内容解析.....	10
3.2.2.3.1 死机信息内容划分.....	10
3.2.2.3.2 基本信息.....	11
3.2.2.3.3 CPU 寄存器信息.....	12



3.2.2.3.4 内存池信息.....	14
3.2.2.3.5 轨迹调度信息.....	15
3.2.2.3.6 任务信息.....	16
3.2.2.3.7 函数调用栈信息.....	17



# 1 SDK 开发环境和 SDK 使用类

本章节主要介绍SDK开发环境搭建以及使用时出现的常见问题处理和解决办法。

## 📖 说明

- Hi3861V100、Hi3861LV100 SDK开发环境的搭建及使用请参见《Hi3861V100 / Hi3861LV100 SDK开发环境搭建 用户指南》、《Hi3861V100 / Hi3861LV100 第三方软件移植指南》。
- 本章节中图例是为说明问题使用的测试代码，与实际使用版本有差异，以构建脚本要求版本为准。

## 1.1 SDK开发环境问题

### 1.2 SDK使用问题

## 1.1 SDK 开发环境问题

### 1.1.1 Python 库安装失败

#### 1.1.1.1 Python 库版本与 Python 版本冲突

#### 问题描述

pycryptodome工具库与Python版本不匹配时会出现如图1-1所示打印。

图 1-1 Python 库版本不匹配示例

```
/tools# pip3 install pycryptodome-3.9.4-cp27-cp27mu-manylinux1_x86_64.whl  
pycryptodome-3.9.4-cp27-cp27mu-manylinux1_x86_64.whl is not a supported wheel on this platform.
```

#### 问题分析

pycryptodome库版本与Python版本不匹配，图1-1中pycryptodome版本是适配python2.7的(cp27)，使用pip3无法安装。



## 解决方法

从工具官网查询与当前Python版本匹配的工具库，下载之后重新安装。

### 1.1.1.2 安装工具库时 Linux 系统相关工具缺失

#### 问题描述

一些Python库的安装需要Linux系统工具的支撑，Python库依赖的工具如果缺失，会引起安装失败。安装失败示例如图1-2所示。

图 1-2 工具缺失导致 Python 库安装失败示例

```
~$ sudo pip3.7 install pycryptodome-3.9.4-cp37-cp37m-manylinux1_x86_64.whl
Exception:
Traceback (most recent call last):
  File "/usr/local/lib/python3.7/site-packages/pip/_internal/cli/base_command.py", line 179, in main
    status = self.run(options, args)
  File "/usr/local/lib/python3.7/site-packages/pip/_internal/commands/install.py", line 255, in run
    with self._build_session(options) as session:
  File "/usr/local/lib/python3.7/site-packages/pip/_internal/cli/base_command.py", line 93, in _build_session
    insecure_hosts=options.trusted_hosts,
  File "/usr/local/lib/python3.7/site-packages/pip/_internal/download.py", line 344, in __init__
    self.headers["User-Agent"] = user_agent()
  File "/usr/local/lib/python3.7/site-packages/pip/_internal/download.py", line 108, in user_agent
    zip(["name", "version", "id"], distro.linux_distribution()),
  File "/usr/local/lib/python3.7/site-packages/pip/_vendor/distro.py", line 120, in linux_distribution
    return _distro.linux_distribution(full_distribution_name)
  File "/usr/local/lib/python3.7/site-packages/pip/_vendor/distro.py", line 675, in linux_distribution
    self.version(),
  File "/usr/local/lib/python3.7/site-packages/pip/_vendor/distro.py", line 735, in version
    self.lsb_release_attr('release'),
  File "/usr/local/lib/python3.7/site-packages/pip/_vendor/distro.py", line 892, in lsb_release_attr
    return self._lsb_release_info.get(attribute, '')
  File "/usr/local/lib/python3.7/site-packages/pip/_vendor/distro.py", line 550, in __get__
    ret = obj._dict[self._fname] = self._f(obj)
  File "/usr/local/lib/python3.7/site-packages/pip/_vendor/distro.py", line 998, in _lsb_release_info
    stdout = subprocess.check_output(cmd, stderr=devnull)
  File "/usr/local/lib/python3.7/subprocess.py", line 395, in check_output
    **kwargs).stdout
  File "/usr/local/lib/python3.7/subprocess.py", line 487, in run
    output=stdout, stderr=stderr)
subprocess.CalledProcessError: Command '['lsb_release', '-a']' returned non-zero exit status 1.
Traceback (most recent call last):
  File "/usr/local/bin/pip3.7", line 10, in <module>
    sys.exit(main())
  File "/usr/local/lib/python3.7/site-packages/pip/_internal/_init_.py", line 78, in main
    return command.main(cmd_args)
  File "/usr/local/lib/python3.7/site-packages/pip/_internal/cli/base_command.py", line 228, in main
    timeout=min(5, options.timeout)
  File "/usr/local/lib/python3.7/site-packages/pip/_internal/cli/base_command.py", line 93, in _build_session
    insecure_hosts=options.trusted_hosts,
  File "/usr/local/lib/python3.7/site-packages/pip/_internal/download.py", line 344, in __init__
    self.headers["User-Agent"] = user_agent()
```

#### 问题分析

Pycryptodome工具库的安装过程需要使用Linux中lsb\_release工具查询Linux的版本，lsb\_release工具的缺失会直接导致Python工具库安装失败。图1-2中由于Linux系统未安装lsb\_release工具导致安装失败。

## 解决方法

在Linux下安装缺失的工具。ubuntu安装lsb\_release命令“sudo apt-get install lsb-core -y”，安装后在Linux终端独立运行命令“lsb\_release -a”，查看命令是否存在和返回结果是否正确。

### 1.1.2 SCons 版本错误





### 1.1.2.1 SCons 版本与 Python 版本冲突

#### 问题描述

SCons版本太低与Python版本不匹配，如图1-3所示。

图 1-3 SCons 版本不匹配示例 1

```
Removed build/scripts/__pycache__/scons_app.cpython-37.pyc
Removed build/scripts/__pycache__/scons_env_cfg.cpython-37.pyc
Removed build/scripts/__pycache__/scons_utils.cpython-37.pyc
Removed directory build/scripts/__pycache__
Removed tools/nvtool/__pycache__/build_nv.cpython-37.pyc
Removed directory tools/nvtool/__pycache__
scons: done cleaning targets.
AttributeError: '_Environ' object has no attribute 'has_key':
  File "/usr/lib/scons/SCons/Script/Main.py", line 1376:
    exec main(parser, values)
  File "/usr/lib/scons/SCons/Script/Main.py", line 1339:
    main(parser)
  File "/usr/lib/scons/SCons/Script/Main.py", line 1111:
    if os.environ.has_key('DH_INTERNAL_OPTIONS'):
                        :~/Hi3861V100R001C01SPC020B020$
```

#### 问题分析

SDK中SCons版本要求3.0.1以上，该SCons版本适配python3，python3语法中不存在has\_key函数，图1-3中SCons版本是适配python2的，不符合编译要求。

#### 解决方法

运行“scons -v”命令，查看版本是否匹配，卸载旧版本SCons，安装对应版本。

### 1.1.2.2 构建脚本中对版本描述与实际版本冲突

#### 问题描述

构建脚本中使用SCons版本与实际安装的版本不一致，出现如图1-4所示错误。

图 1-4 SCons 版本不匹配示例 2

```
scons: Reading SConscript files ...
Python 3.9 or greater required, but you have Python 3.7.3
      :~/Hi3861/new$ 示例代码，实际应该使用3.7+
-bash: fg: current: no such job
      :~/Hi3861/new$ vim SConstruct

[1]+  Stopped                  vim SConstruct
      :~/Hi3861/new$ 示例代码，实际应该使用3.0.1
scons: Reading SConscript files ...
SCons 3.1 or greater required, but you have SCons 3.0.1
```



## 问题分析

SDK使用SCons进行构建，构建脚本中会有SCons版本的选择，安装的SCons版本不一致会导致脚本无法正常运行，需按要求安装环境。

## 解决方法

卸载旧版本SCons，安装SCons构建脚本指定SCons版本。

### 1.1.3 依赖包问题

#### 问题描述

依赖包版本与实际使用版本不一致，如图1-5所示。

图 1-5 pycryptodome 版本与 Python 版本不匹配示例

```
scons: Reading SConscript files ...
3.7.3 (default, Dec 27 2019, 03:17:52)
[GCC 5.4.0 20160609]
ModuleNotFoundError: No module named 'Crypto':
  File "/home/xubinhua/Hi3861V100R001C01SPC020B030/SConstruct", line 7:
    from scripts import common_env, scons_utils, scons_app, scons_env_cfg, pkt_builder
  File "/home/xubinhua/Hi3861V100R001C01SPC020B030/build/scripts/pkt_builder.py", line 12:
    import make_upg_file as MAKE_IMAGE
  File "/home/xubinhua/Hi3861V100R001C01SPC020B030/build/scripts/make_upg_file.py", line 13:
    from Crypto.Hash import SHA
```

## 问题分析

Pycryptodome未安装，或者安装的版本与Python版本不匹配而引发的问题时常发生，因为Linux系统允许安装多个Python版本，依赖包根据不同Python版本安装后，不会共享依赖包。用户在安装依赖包时，容易使用错误的Python版本安装。

## 解决方法

SDK中依赖包需要支持python3.7版本，确认已安装依赖包与编译要求的Python版本相匹配。如果没有安装，请使用python3.7安装。

## 1.2 SDK 使用问题

### 1.2.1 Python 版本错误

#### 问题描述

引发报出Python版本错误的因素有多种，例如：“[1.1 SDK开发环境问题](#)”中介绍的安装库版本不匹配问题。以下提供一些常用的排查思路和解决方法。

## 解决方法

**步骤1** 通过“python -v”查看默认Python版本（要求为python3.7）。

- 如果没有安装python3.7，需安装3.7。



- 如果无法修改默认Python默认使用版本，需在个人目录下，建立python3.7的软链接（例如：~/bin），将此目录添入“PATH”系统参数中，并且此目录路径排在PATH最前，重新登陆终端，再次查看Python版本。

**步骤2** 再次运行SCons，查看打印出的Python版本号。

----结束

## 1.2.2 未指定编译器

### 问题描述

SDK编译前会检查编译器是否已经指定正确。出现错误的情况有：

- 未将编译器路径加到\$PATH中，或者指定不正确。
- \$PATH指定了多款编译器路径，导致检查出错。
- 编译器未完整安装。编译器bin目录中不包含“riscv32-unknown-elf-gcc”。

如果编译器指定不正确，会出现如[图1-6](#)所示的提示。

图 1-6 编译器异常示例

```
SconsBuildError: ===== COULD NOT FIND COMPILER! =====:
File "/home/textuser/Hi3861V100R001C01SPC020B030/SConstruct", line 10:
    env_cfg = scons_env_cfg.SconsEnvCfg()
File "/home/textuser/Hi3861V100R001C01SPC020B030/build/scripts/scons_env_cfg.py", line 13:
    (colors['red'], colors['end']))
```

### 解决方法

检查环境变量\$PATH，正确设置编译器路径。

编译器检查有以下条件：

- 路径中有“hcc\_riscv32”。
- 路径以“bin”或“bin/”结尾。
- bin目录中需包含“riscv32-unknown-elf-gcc”。
- 尽量勿将多个版本的编译器放在同一个满足以上三个条件的路径下，因为系统默认使用第一个被找到的编译器，可能与所需要版本不匹配。

常用shell命令示例：

```
echo $PATH          查询环境变量$PATH已设置路径
#export PATH=$PATH:/absolute_path 设置环境变量$PATH
```



# 2 工具类

本章节主要介绍各工具使用时出现的常见问题和解决办法。

## 说明

Hi3861V100、Hi3861LV100工具的使用方法请参见《Hi3861V100 / Hi3861LV100 HiBurn工具使用指南》，请按照文档提示安装和使用工具。

### [2.1 HiBurn工具常见问题](#)

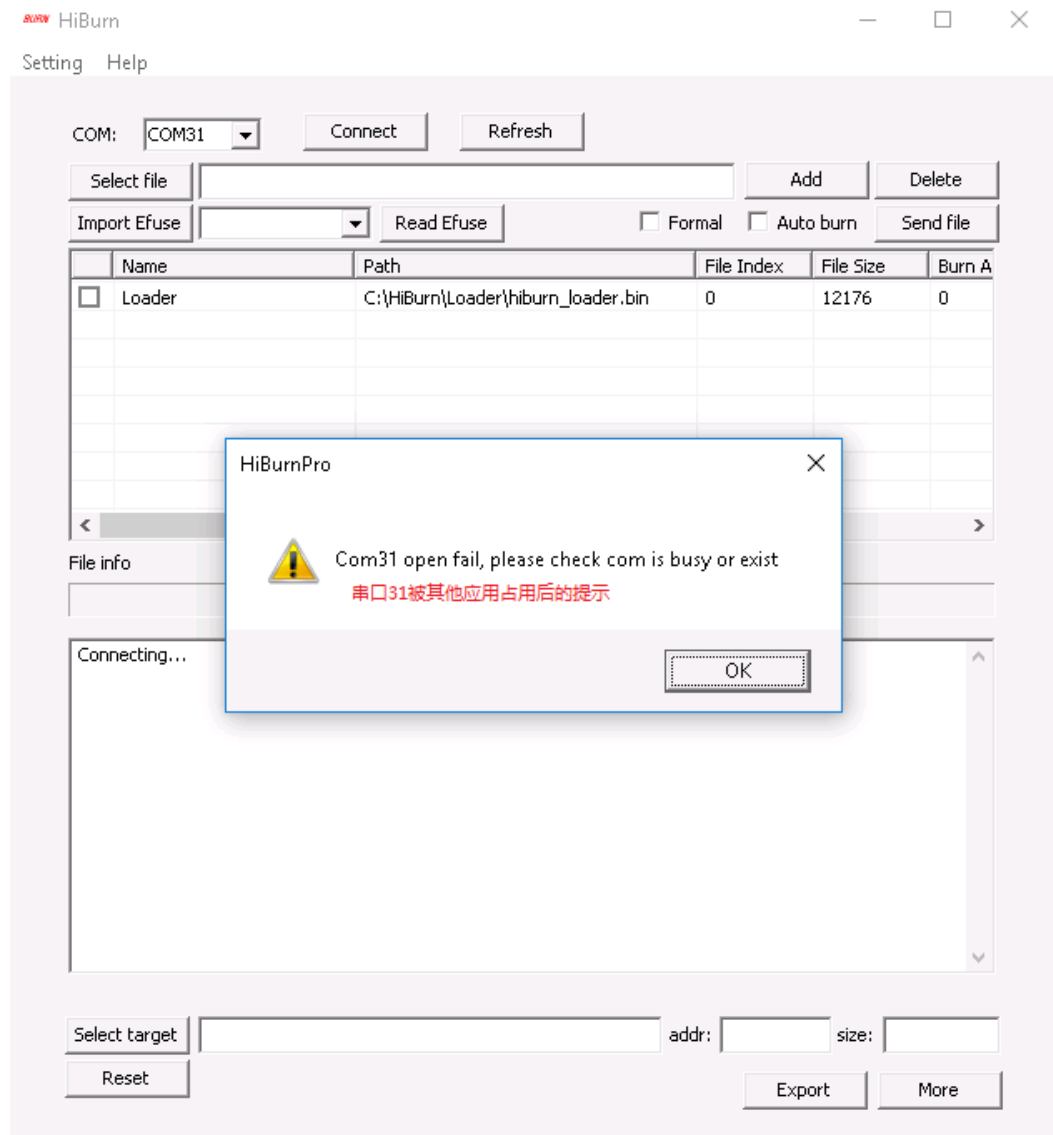
## 2.1 HiBurn 工具常见问题

### 2.1.1 HiBurn 工具与串口连接问题

#### 问题描述

HiBurn工具使用过程由于硬件未连接、串口驱动未安装、串口选择不正确或工具使用不当导致HiBurn工具无法与串口连接。串口被占用示例如[图2-1](#)所示。

图 2-1 HiBurn 工具串口被占用示例



## 解决方法

以下提供了一些常用的解决思路和方法：

- 通过拔插单板，在设备管理器中检查硬件与PC的连接情况。
- 在“设备管理器→端口”查看串口驱动的安装情况，并选择正确的端口。
- 通过HiBurn工具检查串口是否被其他工具占用。如果串口被占用，关闭占用该串口的工具，点击“Refresh”后再进行端口选择和连接。
- 按单板复位键后，HiBurn工具打印窗口有打印表示串口已经连接正常。



## 2.1.2 HiBurn 工具与串口连接后打印窗口一直打印“errno”

### 问题描述

下载过程中打断或断电导致下载失败时，重新连接后多次按单板复位键，会进入下载失败的错误程序中。具体现象如[图2-2](#)所示。

图 2-2 串口窗口错误打印示例

```
errno=0x3605  
errno=0x3613  
errno=0x3605  
errno=0x3613  
errno=0x3605  
errno=0x3613  
errno=0x3605  
errno=0x3613
```

### 解决方法

单板重新插拔上电后，连接HiBurn。



# 3 系统异常类

本章节主要介绍异常触发时看门狗相关问题和异常导致死机信息打印的相关问题。

## 3.1 看门狗常见问题

### 3.2 死机信息及死机问题

## 3.1 看门狗常见问题

### 3.1.1 看门狗简介

看门狗主要用于防止系统任务异常调度问题。如在某个任务或中断中运行过久、不能在规定时间内喂狗，可以被认定为异常，引起系统主动复位进行保护。

### 3.1.2 看门狗超时时间问题

看门狗的系统默认超时时间为  
PRODUCT\_CFG\_AUTO\_WDG\_RESET\_SYSTEM\_TIMEOUT（定义在hi\_config.h文件）。用户可根据实际场景修改宏  
PRODUCT\_CFG\_AUTO\_WDG\_RESET\_SYSTEM\_TIMEOUT的值定义看门狗超时时间，但自定义时间 $\geq 6500\text{ms}$ 。用户也可调用hi\_watchdog\_set\_timeout，动态设置看门狗超时时间，调用该接口后，看门狗超时时间计数将从0重新开始。

因为系统会在空闲时间自动触发喂狗操作，看门狗超时在多数情况下是由于阻塞导致系统无法进入空闲状态。系统阻塞大致分为两种情况：

- 当系统在任务中阻塞时，看门狗会弹出带有“watchdog isr”字样的死机信息，再经过一段时间后复位系统。
- 当系统在中断中阻塞时，看门狗会弹出中断阻塞的位置，然后直接复位系统。

### 3.1.3 看门狗自定义超时时间失败问题

如果编译过程中，出现看门狗自定义超时时间失败的错误，请检查自定义的时间是否低于6500ms，如低于则会产生编译错误，请设置超过6500ms 的自定义超时时间。

图 3-1 看门狗自定义超时时间失败

```
#error "watchdog timeout value must be more than 6500ms!"
```



### 3.1.4 业务任务频繁触发看门狗

用户所在业务频繁触发看门狗，需要检查是否出现以下情况。

- 任务中出现死循环等异常的长时间阻塞动作。如果出现此情况，需要进行修改。
- 在具体业务场景下出现长时间占用CPU的情况。如果出现此情况，需要考虑加大看门狗超时时间，或适当调用hi\_watchdog\_feed进行喂狗。
- 任务循环体根据收到的消息或事件，进入长时间的循环处理流程。如果出现此情况，需要在循环体中调用hi\_cpup\_load\_check\_proc让出CPU。该函数实现的功能是，当检测到CPU占用率大于95%时，使本任务指定睡眠时间，便于让操作系统调度到其它任务。

## 3.2 死机信息及死机问题

### 3.2.1 死机信息简介

死机信息区主要存储程序运行异常相关信息，例如：栈溢出问题、非法地址访问、踩内存等场景的异常信息。

### 3.2.2 死机问题定位

#### 3.2.2.1 查看导出死机信息

用户可根据实际情况在app\_main中调用hi\_syserr\_record\_crash\_info决定是否存储死机信息，记录死机信息在单板重启后，通过输入“AT+DUMP”查看上次死机记录的信息。

#### 3.2.2.2 死机信息定位步骤

死机问题定位通常需要结合死机信息、map文件、反汇编/asm文件进行分析。

定位时需要综合之前修改内容进行分析，必要时需要结合使用场景（电压、时钟等）确定复现规律，具体定位步骤如下：

- 步骤1** 查看死机信息是否可用，如果是死机现场，则打印的死机信息为本次死机。如果想查看最近一次死机信息，需要输入“AT+DUMP”查看。
- 步骤2** 综合死机信息是否有“watchdog\_isr”字样打印、任务信息中Stack Overflow栈溢出标志是否为1，以及mcause、ccause等CPU寄存器值确定死机类型。
- 步骤3** 依据mepc查看反汇编/asm文件，定位死机所在函数位置。
- 步骤4** 依据轨迹调度信息、函数调用信息确认异常前后位置，异常前后任务情况、异常前后函数上下文，最后加以修改。

----结束

#### 3.2.2.3 死机信息内容解析

##### 3.2.2.3.1 死机信息内容划分

死机信息主要内容如表3-1所示。





表 3-1 死机信息列表

成员	描述
版本号	当前版本SDK软件版本号，例如： kernel_ver:Hi3861V100 V100R001C00SPC010。
异常汇总信息	发生异常时的task名称、taskID、task stack size、异常类型。
CPU寄存器信息	发生异常时的CPU寄存器值，包括mepc、mcause、ccause等寄存器。
内存池信息	发生异常时内存池大小、峰值、当前使用大小，内存申请失败次数等。
任务信息	发生异常时所属的任务信息，包括任务栈名称、状态、ID、栈指针、栈顶栈底等。
看门狗信息	发生异常时的看门狗信息，如果是看门狗复位，则PC触发看门狗异常时的指令运行地址。
轨迹调度信息	最近的轨迹调度信息，显示了系统调度轨迹，例如：中断、任务等。
函数调用栈信息	任务栈中的代码执行记录，从上到下表示由近及远的代码执行顺序，可以依据此信息得出异常发生时的程序上下文。

### 3.2.2.3.2 基本信息

基本信息打印了CRC校验值、死机信息长度、日志版本等死机的基本信息，同时用户也可以依据表3-3确认导致系统复位的原因，基本信息内容如表3-2所示。

表 3-2 基本信息表

成员	描述
crc_usable	死机信息CRC校验是否正确（1为正确）。
info_len	死机信息长度。
crc_val	CRC16校验值。
log_ver	日志版本号。
eid	复位类型，具体类型如表3-3所示。
rid	异常类型，具体类型如表3-3所示。
crash_tsec	异常时相对当次启动时间。
boot_ver	安全启动模式下的Boot版本号（如果为非安全模式启动，该值为0）。
type_name	类型名，与eid对应。



eid&rid两者配合使用，分别读出eid和rid的值，然后参照表3-3即可快速判断出此次死机或异常的原因。

例如：eid值为0x7、rid值为0x200，即可判断出此次系统复位是由WiFi模块引起。

表 3-3 eid&rid 描述表

事件描述	eid值	rid值
下电重启或首次上电或硬复位	0x0	0x0
任务中死机	0x1	为mcause寄存器值，详细内容如表3-5所示
中断中死机	0x2	为mcause寄存器值，详细内容如表3-5所示
看门狗任务中死机	0x3	0x0
看门狗中断中死机	0x4	0x0
系统软复位（未知原因）	0x5	0x0
系统软复位（AT命令主动复位）	0x5	0x1
系统软复位（升级复位，双分区升级时当前运行在A区）	0x5	0x2
系统软复位（升级复位，双分区升级时当前运行在B区）	0x5	0x3
系统软复位（WiFi驱动主动复位）	0x5	0x4
系统软复位（用户正常正常复位）	0x5	0x5
系统软复位（用户自定义复位0）	0x5	0x6
系统软复位（用户自定义复位1）	0x5	0x87
系统软复位（AT命令连续多次busy主动复位）	0x5	0x8

### 3.2.2.3.3 CPU 寄存器信息

死机信息打印提供如表3-4所示的CPU寄存器以协助开发人员进行死机原因及死机位置的定位。



表 3-4 死机相关 CPU 寄存器描述

成员	描述
mepc	机器异常程序计数器。当发生异常时，mepc指向导致异常的指令；对于中断，mepc指向中断处理后应该恢复的位置。
mstatus	机器状态寄存器。通过它的最低位判断是否使能中断（0：禁止中断；1：使能中断）。
mtval	机器陷入寄存器。保存地址异常中出错的地址或者发生指令异常的指令本身，对于其他错误，其值为零。
mcause	机器异常寄存器。保存目前异常或者中断的原因，通过查询表3-5得到目前异常或者中断的类型。
ccause	与mcause类似，ccause为mcause的补充说明，对于某些异常通过读取ccause寄存器的内容可以进一步明确异常类型。

表 3-5 mcause&ccause 异常描述表

异常码	mcause异常描述	ccause异常描述
0x80000021	Machine non-standard local interrupt	Not available
0x0	Instruction address misaligned	Not available
0x1	Instruction access fault	Memory map region access fault
0x2	Illegal instruction	AXIM error response
0x3	Breakpoint	AHBM error response
0x4	Load address misaligned	Crossing PMP entries
0x5	Load access fault	System register access fault
0x6	Store/AMO address misaligned	No PMP entry matched
0x7	Store/AMO access fault	PMP access fault
0x8	Environment call from U-mode	CMO access fault
0x9	Environment call from S-mode	CSR access fault
0xa	Reserved	LDM/STMIA instruction
0xb	Environment call from M-mode	ITCM write access fault
0xc	Instruction page fault	Not available
0xd	Load page fault	Not available
0xe	Reserved	Not available



异常码	mcause异常描述	ccause异常描述
0xf	Store/AMO page fault	Not available
> 0xf	Reserved	Not available

当系统主动抛出异常时，CPU通用寄存器会记录具体异常的原因，可以通过死机打印a1寄存器的值查看。

表 3-6 主动抛异常原因解析表

a1寄存器值	异常原因
0x64	栈保护检测异常。
0x65	Os内存释放异常，taskID值大于taskID最大值，非法。
0x66	Os内存释放异常，task未被正确创建。
0x67	Os内存释放异常，task内存并非本次操作申请，由其他task申请。
0x68	Os内存完整性检测异常，taskID值大于taskID最大值，非法。
0x69	Os内存完整性检测异常，task未被正确创建，
0x6a	Os内存完整性检测异常，task内存并非本次操作申请，由其他task申请。
0x6b	Os内存检测异常，栈溢出，栈指针被破坏。

- 当发生栈保护检测异常时，CPU通用寄存器a2会同时保存发生栈检测异常的函数地址，需结合函数地址检测对应函数是否存踩内存和越界写。
- 当发生Os内存释放和申请异常时，大概率为踩内存导致Os内存链表异常。

#### 3.2.2.3.4 内存池信息

内存池信息打印效果如图3-2所示，通过了解内存池的大小、峰值、申请失败次数、已经使用的内存可以确认资源使用是否发生异常。

图 3-2 内存池信息示例

```
*****memory info*****
Pool Addr    = 0xe9b80
Pool Size    = 0x2ea40
Fail Count   = 0x0
Peek Size    = 0x184e4
Used Size    = 0x182644
```



### 3.2.2.3.5 轨迹调度信息

轨迹调度信息打印首先会打印当前调度项current\_item和准备打印的调度项总数目item\_cnt，接下来会打印一个表格，表格包含以下内容：

- Index：调度项目编号。
- TrackType：具体调度项操作，具体类型如表3-7所示。
- TrackID：调度项ID，ID具体内容由调度项类型决定：
  - 中断：保存中断向量。
  - 工作队列：保存工作优先级。
  - 任务：保存任务ID。
  - 事件：该值为0。
- CurTime：事件发生时系统tick数。
- Data1 & Data2：发生事件的PC指针。在调度项类型为任务且具体操作为任务调度时，Data1记录旧任务的PC现场，Data2记录新任务的PC现场。

表 3-7 TrackType 表

调度项类型	具体调度项操作	TrackType
TRACK_ISR	TRAP_INT	0016
TRACK_IRQ	ENABLE_IRQ	0032
	DISABLE_IRQ	0033
TRACK_TASK	CREATE_TSK	0064
	SCHEDULE_TSK	0065
	DELETE_TSK	0066
TRACK_WORKQ	ENQUEUE_WORKQ	0128
	SCHEDULE_WORKQ	0129
	DEQUEUE_WORKQ	0130
TRACK_SWTMR	CREATE_SWTMR	0256
	START_SWTMR	0257
	MODIFY_SWTMR	0258
	SCHDING_SWTMR	0259
	SCHDED_SWTMR	0260
	STOP_SWTMR	0261
	DEL_SWTMR	0262
	DROP_SWTMR	0263
TRACK_MUX	MUX_CREATE	0512



调度项类型	具体调度项操作	TrackType
	MUX_PEND	0513
	MUX_POST	0514
	MUX_DEL	0515
TRACK_SEM	SEM_CREATE	1024
	SEM_PEND	1025
	SEM_POST	1026
	SEM_DEL	1027
TRACK_QUE	QUE_CREATE	2048
	QUE_PEND	2049
	QUE_POST	2050
	QUE_DEL	2051
TRACK_EVENT	EVENT_CREATE	4096
	EVENT_WRITE	4097
	EVENT_READ	4098
	EVENT_CLEAR	4099
	EVENT_DEL	4100

轨迹调度信息示例如图3-3所示，用户可以依据轨迹调度信息检查发生异常前后的运行关系。

图 3-3 轨迹调度信息示例

```
*****track_info*****
current_item:0x1
item_cnt:0xa
Index   TrackType  TrackID  CurTime  Data1  Data2
0001 0016 0007 0x42c 0x496bd4 0x0
0002 0016 0007 0x424 0x3f5a60 0x0
0003 0016 0007 0x425 0x3f5a62 0x0
0004 0016 0007 0x426 0x3f5a62 0x0
0005 0016 0007 0x427 0x3f5a62 0x0
0006 0016 0007 0x428 0x3f5a60 0x0
0007 0016 0007 0x429 0x3f5a64 0x0
0008 0016 0007 0x42a 0x3f5a62 0x0
0009 0016 0007 0x42b 0x3f5a62 0x0
0010 0064 0009 0x42c 0x0 0x0
```

3.2.2.3.6 任务信息

任务信息会较为全面地展示当前异常时执行的任务信息，任务信息包括以下内容：



- Name: 任务名称。
- ID: 任务ID。
- Status: 当前任务状态。
- Stack Peak: 栈使用峰值。
- Stack Size: 任务栈大小。
- SP: 栈指针。当前任务需要切换为其他任务时, 作为恢复现场使用的栈指针; 当异常发生时, 该值会切换为异常栈指针。
- Stack: 任务栈栈顶和栈底指针。
- Real SP: 真实栈指针。异常发生时实际的栈指针, 即SP切换为异常栈之前的栈指针。
- Stack Overflow: 栈溢出标志位。

任务信息示例如图3-4所示。

图 3-4 任务信息示例

```
*****task info*****
Name       : usr_app
ID         = 2
Status     = 0x14
Stack Index = 0x8
Stack Peak = 0x4e8
Stack Size = 0x4000
SP         = 0x11a860
Stack      : 0xec330 to 0xf0380
Real SP    = 0xf02c0
Stack Overflow = 0
```

### 3.2.2.3.7 函数调用栈信息

函数调用栈call stack会显示与异常相关的所有函数调用指令。用户可以根据函数调用栈检查异常发生时函数调用的上下文以方便定位。函数调用栈信息如图3-5所示。

图 3-5 函数调用栈信息示例

```
*****call stack*****
call stack 0 -- 3f78c0 addr:f036c
call stack 1 -- 3f5e24 addr:f037c
*****call stack end*****
```