Senscomm

IOT 闪存工具

软件

命令行界面

用户指南

V1.1

修订历史

版本	日期	修订
V0.1	2023/06/12	初版
V0.2	2023/06/26	安排“开始使用非故障安全设备”的步骤。
V1.0	2023/07/12	V1.0 发布版本
V1.1	2023/07/18	修改 • flash_write 命令描述 • efuse_write 命令描述 增加 • Q&A

第一章 简介

命令行界面 (CLI) 用于编程设备，并且可以在不同的操作系统上执行。为了适应各种情况，它需要参数进行配置或使用配置文件。

为了启用 CLI 通信，设备需要设置为 "从 Uart 启动"。该设备支持不同的启动模式。

启动模式	GPIO3	GPIO2
Boot From Flash	1	0
Boot From Uart	**0**	**1**
Boot From Usb	1	0
Boot From SDIO	0	0

表 1-1: 启动模式

如果设备支持硬件重置，CLI 工具将自动选择启动模式并重置设备。

在以下内容中，CLI 通常被称为 "sctool"。Sctool 主要包括两个部分：

- **可选**: 此部分指的是用于配置或修改整个过程行为的参数。
- **操作**: 此部分包括执行部分，执行以下任务：
  - 信息查询
  - Flash 操作
  - eFuse 操作

整个过程可以分为四个阶段。

第二章 CLI

执行指令: sctool

```
usage: sctool [-h] [-v VERBOSE] [-V] [-p PORT] [-da DA] [-b BAUDRATE]
              [--before {hw_reset,no_reset}] [--manual]
              [--after {hw_reset,sw_reset,no_reset}] [-c CONFIG]

{test_write,test_read,hw_reset,sw_reset,list_ports,chip_id,da_ver,upload_da,flash_read,flash_write,flash_erase,flash_size,flash_chip_erase,efuse_read,efuse_write,efuse_read_bin,efuse_write_bin,efuse_size}
```

选项指南

- -h : 打印帮助信息
- -v : 输出信息的详细级别
  - 0 – 只显示关键消息。
  - 1 – 显示关键和信息消息（默认）。
  - 2 – 显示所有消息。
- -p : 与设备通讯的串口号
- -da : DA 镜像的路径。
- -b : 上传 DA 后通信的新波特率。
- --before: 在前期阶段的动作。
  - hw_reset **(默认)**
  - no_reset
- --after: 在后期阶段的动作。
  - hw_reset
  - sw_reset
  - no_reset **(默认)**
- --manual : 只执行操作。
- -c : 配置文件路径。

为了满足基本的命令需求，命令中应同时包括端口号和 DA。以下是读取"da_ver"的示例命令:

对于 windows 系统 - read da_ver

```
sctool.exe -p COM3 -b 2000000 -da da/da.ram.bin da_ver
```

对于 Ubuntu 18.04系统 - read da_ver

```
./sctool -p /dev/ttyUSB0 -b 2000000 -da da/da.ram.bin da_ver
```

以下示例将使用 Windows 作为模板。对于 Ubuntu 版本，请将“sctool.exe”替换为“./sctool”，并将“COMX”替换为“/dev/ttyUSBX”。

配置文件

为了简化命令并减少操作所需的参数数量，使用配置文件会更加方便。配置文件应为 INI 格式，结构应类似于以下内容：

```
[Settings]
verbose = 1
port = COM3
agent = da/da.ram.bin
baudrate = 2000000
before = hw_reset
after = no_reset
```

在命令行中，通过传递“-c”参数来使用配置文件。

对于 windows 系统 – 读取 "da_ver"

```
sctool.exe -c config.ini da_ver
```

可以将配置文件用作默认值，并在需要时提供覆盖它的选项。

对于 windows 系统 – 使用 COM0 读取 "da_ver"。

```
sctool.exe -c config.ini da_ver -p COM0
```

对于 windows 系统 – 使用 COM0 和波特率 115200 读取 "da_ver"。

```
sctool.exe -c config.ini da_ver -p COM0 -b 115200
```

操作说明

此操作需要某些参数，并且可以添加额外的可选参数，如“--verify”。为提供一个简化的示例，以下模板假定您正在使用 Windows 系统。如果您使用的是 Ubuntu，请相应地将 "sctool.exe" 替换为 "./sctool"。

测试

test_write <SIZE>

写入特定大小以测试通信。

```
sctool.exe -c config.ini test_write 512
```

test_read <SIZE>

读取特定大小以测试通信。

```
sctool.exe -c config.ini test_read 512
```

通用操作

chip_id

读取设备的芯片 ID

```
sctool.exe -c config.ini chip_id
```

da_ver

读取 da 版本

```
sctool.exe -c config.ini da_ver
```

upload_da

上东上传 da

```
sctool.exe -c config.ini upload_da
```

重置操作

hw_reset

执行硬件重置（此功能需要硬件支持）

```
sctool.exe -c config.ini hw_reset
```

sw_reset

执行软件重置（设备必须加载 DA）

```
sctool.exe -c config.ini upload_da
sctool.exe -c config.ini sw_reset
```

其他操作

list_ports

列出用于 UART 的可用端口（此方法不需要 config.ini）

```
sctool.exe list_ports
```

闪存操作

flash_read <ADDR> <SIZE> <FILENAME>

从 <ADDR> 读取大小为 <SIZE> 的闪存，并将其保存到 <FILENAME>。

由于闪存映射方法，闪存的起始地址为 "0x80000000"。(0x80000000 在闪存地址上意味着 0x00000000)

```
sctool.exe -c config.ini flash_read 0x80000000 0x2000 boot.bin
```

flash_write <ADDR> <FILENAME> [--verify]

从 <ADDR> 开始写入闪存，数据内容来自 <FILENAME>。

由于闪存映射方法，闪存的起始地址为 "0x80000000"。(0x80000000 在闪存地址上意味着 0x00000000)

```
sctool.exe -c config.ini flash_write 0x80000000 boot.bin
```

--verify: 写入数据后，读回并使用 <FILENAME> 进行验证。

```
sctool.exe -c config.ini flash_write 0x80000000 boot.bin --verify
```

警告: flash_write 将自动执行擦除操作，擦除块为 4KB 对齐。

flash_erase <ADDR> <SIZE>

从 <ADDR> 开始擦除大小为 <SIZE> 的闪存。

```
sctool.exe -c config.ini flash_erase 0x80000000 0x2000
```

警告 : 请注意，擦除操作要求大小（<SIZE>）与 4KB 对齐。这意味着实际大小应增加 0x1000。

flash_size

获取闪存大小 (Bytes)。

```
sctool.exe -c config.ini flash_size
```

flash_chip_erase

执行芯片擦除。

```
sctool.exe -c config.ini flash_chip_erase
```

EFUSE 操作

efuse_read <START_BIT> <BIT_WIDTH>

从 <START_BIT> 读取 efuse 数据，位宽为 <BIT_WIDTH>。

```
sctool.exe -c config.ini efuse_read 0 8
```

efuse_write <START_BIT> <BIT_WIDTH> <HEX_DATA>

从 <START_BIT> 写入 efuse 数据，位宽为 <BIT_WIDTH>，数据为 <HEX_DATA>。

```
sctool.exe -c config.ini efuse_write 0 64 0x0123456789ABCDEF
```

警告: 请注意，使用 efuse_write 功能将永久更改 efuse 的行为。efuse_write 只能将数据从“0”更改为“1”。

efuse_read_bin <FILENAME>

读取整个 efuse 并将其保存到 <FILENAME>。

```
sctool.exe -c config.ini efuse_read_bin efuse.bin
```

efuse_write_bin <FILENAME> [--verify]

使用来自 <FILENAME> 的内容编写整个 efuse 数据。

```
sctool.exe -c config.ini efuse_write_bin efuse.bin
```

--verify: 写入 efuse 数据后，读回并使用 <FILENAME> 进行验证。

```
sctool.exe -c config.ini  efuse_write_bin efuse.bin --verify
```

警告**:** efuse_write_bin 只能将数据从“0”更改为“1”。

efuse_size

获取 efuse 大小（位）。

```
sctool.exe -c config.ini efuse_size
```

第三章 示例

FLASH 操作

XIP 启动

```
sctool.exe -c config.ini --after hw_reset flash_write 0x80000000 image/nuttx.scmboot.xip.bin
```

Nuttx 启动

```
sctool.exe -c config.ini --after hw_reset flash_write 0x80040000 image/nuttx.mcuboot.xip.bin
```

Wise

```
sctool.exe -c config.ini --after hw_reset flash_write 0x80140000 image/wise.bin
```

手动烧录多扇区

```
sctool.exe -c config.ini --before=hw_reset --after=no_reset upload_da
sctool.exe -c config.ini --manual flash_write 0x80000000 image/nuttx.scmboot.xip.bin
sctool.exe -c config.ini --manual flash_write 0x80040000 image/nuttx.mcuboot.xip.bin
sctool.exe -c config.ini --manual flash_write 0x80140000 image/wise.bin
sctool.exe -c config.ini hw_reset
```

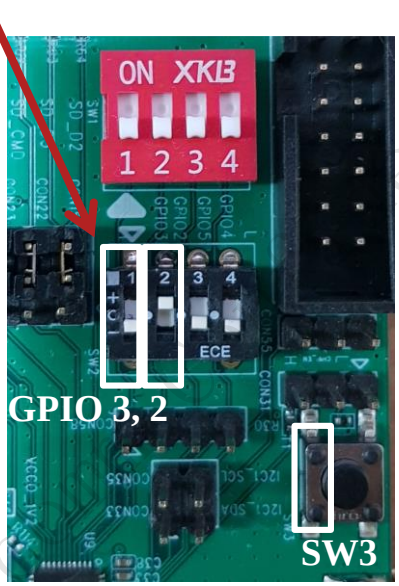
对于不支持安全恢复机制的设备：

如果您的设备不支持安全恢复机制，您需要逐步使设备进入“通过 UART 启动”并进行编程。请按照以下说明操作。

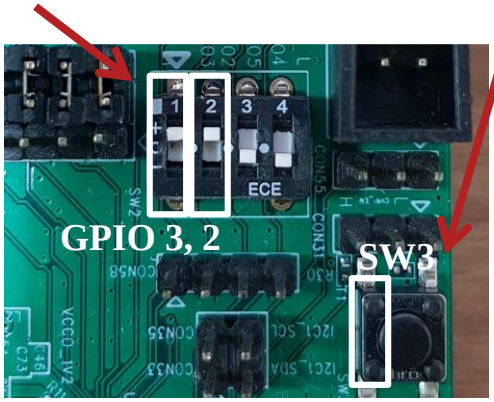
如果您的设备不支持安全恢复机制，那么您需要经过一系列的步骤来进入“通过 UART 启动”模式并进行编程。在这一章中，您将找到关于如何为这类设备进行故障排除，并成功初始化 UART 通讯的启动过程的详细说明。请按照以下步骤进行，以确保编程过程的顺利和高效。

逐步指南：

- **步骤 1:** 将 SW2 切换至“0100”，以实现“通过 UART 启动”。



- **步骤 2:** 打开目标设备电源。
- **步骤 3:** 执行 CLI 操作，它将等待设备进入 UART 模式。
  例如: sctool.exe -c config.ini flash_read 0x80000000 0x2000 boot.bin
- **步骤 4:** 触发 SW3 按钮进行硬件重置以进入 UART 模式并等待命令完成。
- **步骤 5:** 将 SW2 恢复为“从 Flash 启动”，然后触发 SW3 进行重启。



第四章 常见问题

Q: 出现错误信息 : “Error: Could not open serial port COMn” on Windows or “Error: Could not open serial port /dev/usbttyn” on Linux

A: 看起来操作系统没有检测到 COM 端口，您可以输入以下命令来检查可用的 COM 端口并选择其中一个。

```
sctool.exe list_ports
Available Ports:
   COM1
   COM4
```

Q: 出现错误信息: “Error: Serial Port Read Timeout!”

A: COM 端口已经被检测到，但通信失败。请检查您是否已经将 EVB/EVK 切换至“通过 UART 启动”模式。在设置“通过 UART 启动”模式并触发重置按钮后，您将在终端上看到 "0x00000000 CCC..."。

Q: 为什么在输入 'sctool.exe xxx' 命令后需要立即点击重置按钮?

A: Flash 工具需要目标设备上的兼容 DA 镜像来处理相关命令。“通过 UART 启动”模式可以允许 sctool.exe 上传 DA。如果设备不在“通过 UART 启动”模式，命令将每秒检查一次，直到等待10秒。这就是为什么我们需要在10秒内点击重置按钮，进入“通过 UART 启动”模式，让 sctool.exe 完成其命令的原因。

Q: 在 Linux 电脑上，如何确认用于刷写的镜像端口，是 /dev/ttyUSB0 还是 /dev/ttyUSB1?

A: 在 Linux 上，它会将第一个 USB-UART 设置为 /dev/ttyUSB0，第二个设置为 /dev/ttyUSB1。

为了确定哪个端口正在用于刷写图像，您可以使用 minicom 来检查其中哪个在“通过 UART 启动”模式下持续输出 "CCC…"。以下是示例。