



SCM1612

Wi-Fi 6 和 BLE 5 低功耗 SoC

SDK 入门指南

文档版本 1.3
发布日期 2024-05-27

联系方式

速通半导体科技有限公司 (www.senscomm.com)
江苏省苏州市工业园区苏州大道西 2 号国际大厦 303 室
销售或技术支持, 请发送电子邮件至
support@senscomm.com

免责声明和注意事项

本文档仅按"现状"提供。速通半导体有限公司保留在无需另行通知的情况下对其或本文档中包含的任何规格进行更正、改进和其他变更的权利。

与使用本文档中的信息有关的一切责任，包括侵犯任何专有权利的责任，均不予承认。此处不授予任何明示或暗示、通过禁止或其他方式对任何知识产权的许可。

本文档中的所有第三方信息均按"现状"提供，不对其真实性和准确性提供任何保证。

本文档中提及的所有商标、商号和注册商标均为其各自所有者的财产，特此确认。

© 2024 速通半导体有限公司。保留所有权利。

Senscomm Confidential

版本历史

版本	日期	描述
1.3	2024-05-27	更新 UART 信息
1.2	2023-09-09	完善 EV 板指南的描述
1.1	2023-08-15	格式修改
1.0	2023-08-04	1.0 发布
0.1	2023-07-11	初稿

目录

版本历史.....	3
1 简介.....	5
2 设置开发环境.....	6
2.1 安装 Linux 软件包.....	6
2.2 工具链.....	6
2.3 Python 和软件包.....	7
2.4 在 Windows 上安装 AICE 驱动.....	7
2.5 终端应用程序.....	7
3 构建固件.....	8
3.1 目录结构.....	8
3.2 构建.....	8
3.3 修改配置.....	9
3.4 安全引导.....	9
3.5 闪存加密.....	10
4 构建主机(Host)驱动程序.....	11
4.1 构建.....	11
4.1.1 配置文件选择与设置.....	11
4.1.2 编译驱动 sncmfmac.ko.....	11
4.1.3 编译应用程序.....	12
5 下载镜像.....	13
5.1 在 RAM 上运行引导程序:.....	16
5.2 烧录 XIP 引导程序.....	16
5.3 烧录主固件.....	17
5.4 自动运行固件.....	17
5.5 仅更新固件.....	18
6 调试.....	19
6.1 JTAG 调试使用 OpenOCD.....	19
6.1.1 概述.....	19
6.1.2 设置调试环境.....	20
6.1.3 远程调试.....	21

1 简介

本文档介绍了基础知识：如何设置开发环境、构建 SDK 和托管驱动程序（如果需要），以及如何下载和运行固件。

由速通半导体提供的 SDK 文件包含以下目录。

- 文档
- 软件
- 工具链

2 设置开发环境

为构建 scm1612 设备固件，推荐在 64 位 Linux PC 上进行。建议使用 Ubuntu 20.04 或更高版本。在接下来的章节中，假设用户使用的是 Ubuntu。

2.1 安装 Linux 软件包

为了进行基本构建，请安装以下软件包。

```
$ sudo apt install build-essential libncurses-dev  
$ sudo apt-get install libevent-dev libnl-3-dev libnl-genl-3-dev
```

2.2 工具链

将工具链解压缩到 “/opt/” 目录中。

```
$ sudo tar xvf nds32le-elf-mculib-v5.tar.gz -C /opt/
```

默认情况下，SDK 假定工具链路径为 “/opt/nds32le-elf-mculib-v5/bin/”。如果工具链被解压到其他位置，则在继续构建之前，必须使用 “make menuconfig” 相应地更改路径。

```
(/opt/nds32le-elf-mculib-v5/bin/) Cross toolchain path  
(rv32imac) -march option  
(ilp32) -mabi option  
(49) number of interrupt lines  
(2) number of SW interrupt lines  
*- Device region defined  
(32) (I/D)-Cache line size  
[ ] Support prioritized interrupt  
(2048) interrupt stack size (bytes)  
*- Instruction cache  
[ ] Data cache  
[ ] Support EXEC.IT optimization  
(-Os -g3) GCC optimization option  
[*] Enable core dump  
[ ] Compile the Wise with frame pointers
```

2.3 Python 和软件包

构建过程中的某些步骤涉及运行 Python 脚本。安装以下 Python 软件包。

```
$ sudo apt install python3-pip
$ pip install pycryptodome
$ pip install imgtool
```

由于这些软件包通常安装在 `/home/{user}/.local/bin` 目录下，所以必须将该路径添加到 `PATH` 变量中。

编辑 `~/.bashrc` 文件，并添加以下行。

```
export PATH=$PATH:~/.local/bin
```

另一种方法是在目录下创建一个名为 `'bin'` 的文件夹，Linux 发行版会自动将 `'bin'` 和 `'./local/bin'` 添加到 `PATH` 中。

2.4 在 Windows 上安装 AICE 驱动

为使用 JTAG 进行调试，请确保已连接 AICE 适配器。

- 安装 AICE 驱动程序

2.5 终端应用程序

需要一个串行终端应用程序来：

- 通过 UART 下载固件
- 查看控制台日志

3 构建固件

3.1 目录结构

SDK 的顶级目录包含以下子目录。

目录	描述
api	Senscomm Wi-Fi APIs
app	示例应用程序
configs	默认配置
hal	硬件抽象层
include	需要包含的头文件
kernel	带有 FreeRTOS 和 FreeBSD 的内核
lib	可能使用到的库模块
prebuilt	使用预定义配置构建的库
scripts	与构建相关的脚本

SCM1612 芯片内部集成了一个 ROM，其中包含一些常用的软件组件，如 C 库、操作系统、WLAN 驱动、网络库和 BLE 控制器相关接口。部分软件组件已经集成在 ROM 中并已具体实现，而在发布的 SDK 中，这些组件仅提供了头文件。

3.2 构建

使用 'make distclean' 命令以清除上次构建的残留文件。在更改任何配置之前，最好先清理文件。

在配置目录中选择一个合适的配置文件作为构建的起点。

构建一个在 RAM 中运行的引导加载程序。

```
$ make distclean
$ make scm1612s_bl_ram_defconfig
$ make
```

这将产生 'wise.scmboot.ram.bin'

构建一个用于固件烧录的引导加载程序


```
$ make distclean
$ make scm1612s_bl_defconfig
$ make
这将产生 `wise.scmboot.bin`
```

构建 Standalone 模式的固件

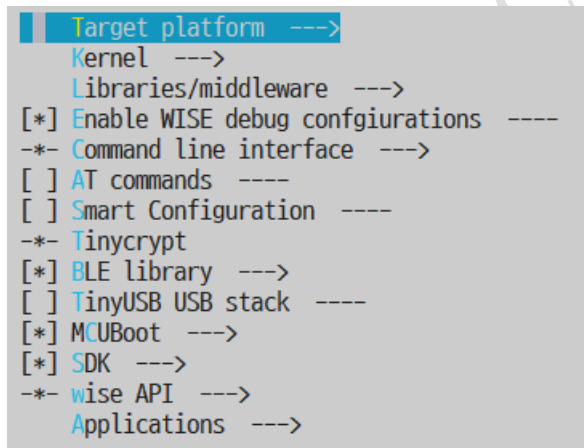
```
$ make distclean
$ make scm1612s_defconfig
$ make
这将生成 `wise.mcuboot.bin`
```

构建过程完成后，将生成 `wise.xxxx.bin` 文件。二进制文件名根据所用的配置可能会稍有不同。

3.3 修改配置

构建系统基于传统的 `Kconfig` 和 `Kbuild`。如果需要更改任何配置选项，请运行 `'make menuconfig'`，并导航到需要修改的选项。

```
$ make menuconfig
```



```
Target platform --->
Kernel --->
Libraries/middleware --->
[*] Enable WISE debug configurations ----
-* Command line interface --->
[ ] AT commands ----
[ ] Smart Configuration ----
-* Tinycrypt
[*] BLE library --->
[ ] TinyUSB USB stack ----
[*] MCUBoot --->
[*] SDK --->
-* wise API --->
Applications --->
```

3.4 安全引导

支持安全引导，只允许启动经过验证的固件。
如需更多信息，请联系 [Senscomm](#) 技术支持。

3.5 闪存加密

支持闪存加密以保护固件。即使将固件烧录到另一个 SCM1612 芯片，该固件也将无法使用。必须使用相应的安全凭据写入 eFuse。

有关更多信息，请联系 [Senscomm](https://www.senscomm.com).

Senscomm Confidential

4 构建主机(Host)驱动程序

本节仅适用于希望通过 SDIO 或 USB 将 SCM1612 作为 Wi-Fi 接口连接到主机平台的用户。

若 SCM1612 与主机平台连接，必须在主机上安装并运行相应的内核驱动程序。

4.1 构建

4.1.1 配置文件选择与设置

```
cd xiaohu-ax/  
cp configs/cfg_XXX.mk cfg.mk
```

由于支持多种平台和配置模式，如 SDIO OOB 模式、SDIO INT 模式、USB 模式等。用户要依据平台与需求首先选择一个配置文件进行编译。系统默认提供配置文件保存在 configs/目录下：

配置文件	平台/SDIO 模式
cfg_sdio_normal_fullhan.mk	Fullhan, 4 bit mode interrupt
cfg_sdio_normal_goke.mk	Goke, 4 bit mode interrupt
cfg_sdio_normal.mk	X86/X64, 4 bit mode interrupt
cfg_sdio_polling.mk	Linux PC, Polling mode
cfg_sdio_oob_int_goke.mk	Goke, OOB interrupt mode
cfg_usb.mk	X86/X64 USB 模式

4.1.2 编译驱动 sncmfmac.ko

虽然 ARCH 和 CROSS_COMPILE 在文件中已经配置好，但参数'KDIR'需要根据用户环境进行明确指定。其中，KDIR=/home/apache/page/xxx 是指定的 kernel 所在位置。

```
make KDIR=/home/apache/page/linux-4.9
```

如果没有指定'KDIR'，系统则会使用默认的内核 KDIR，这将编译出的驱动将适用于 X86/X64 平台上运行。

```
make
```

4.1.3 编译应用程序

```
make KDIR=/home/apache/page/linux-4.9 apps
```

目前有两种类型的应用程序：

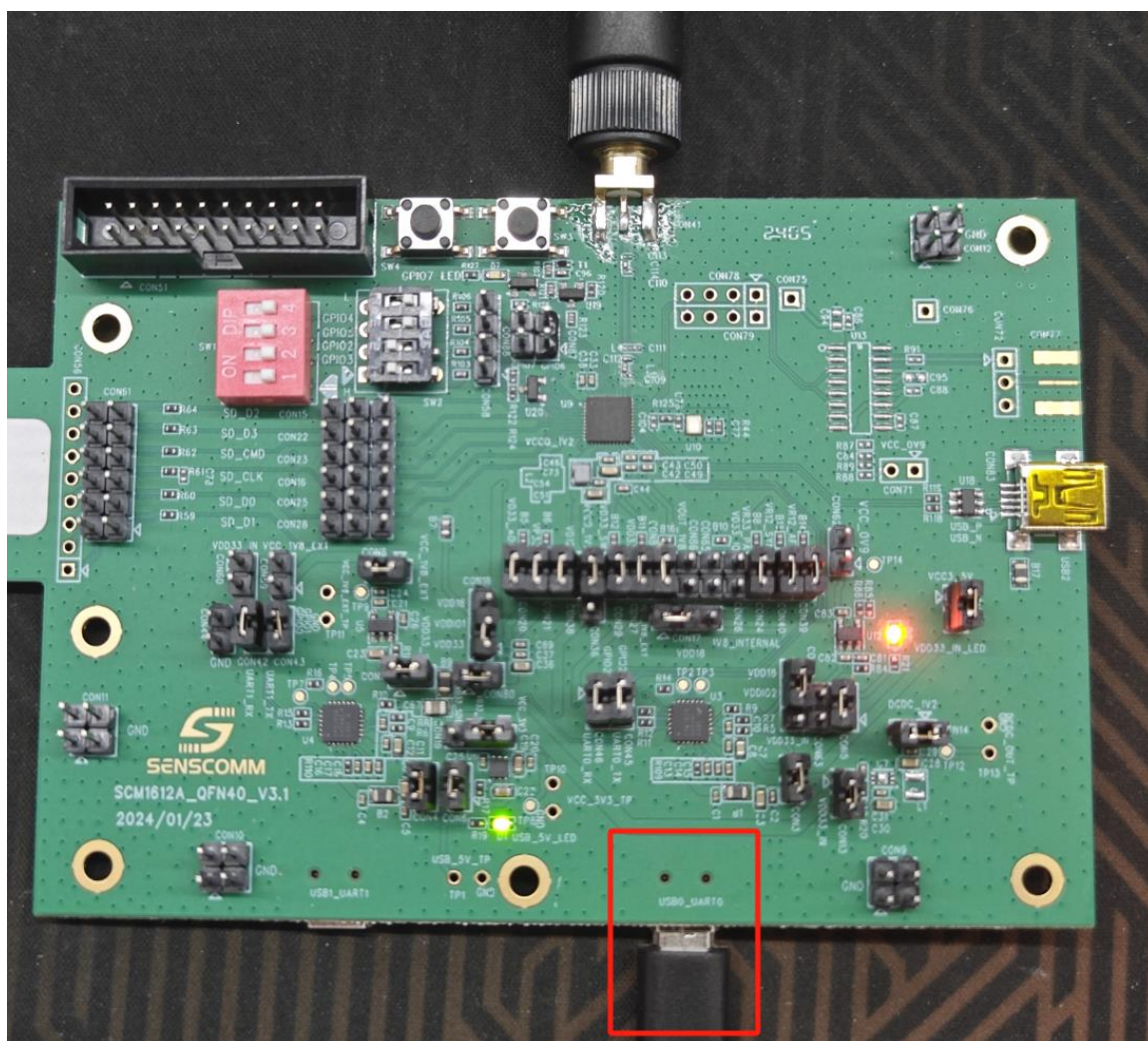
- 'sncm_cmd': 通过主机命令进行 Wi-Fi 配置和控制。
- 'sncm_chn'（也称为 ScmChannel）：Wi-Fi 配置和控制 在 SCM1612 侧完成，通过 ScmChannel 获取 Wi-Fi 连线信息。
 - 'sample_link'主要用于同步 SCM1612 侧网络节点的 Mac 地址、IP 地址等信息。
 - 'sample_cli':主要用于发送客户自定义的信息

5 下载镜像

SCM1612 的启动模式取决于 GPIO 的设置。

启动模式 1 (GPIO3)	启动模式 0 (GPIO2)	启动模式描述
1	1	FLASH (从闪存启动)
0	1	UART (从 UART 下载固件)
1	0	USB (从 USB 下载固件)
0	0	SDIO (从 SDIO 下载固件)

- 请注意，在 SCM2010_QFN40_V2.0 EVB 板上，SW2 的标识存在错误，将 GPIO2 和 GPIO3 的标签颠倒了。
- 在 Windows 电脑上，需要使用支持 Ymodem 传输协议的工具，我们推荐使用 Tera Term。
- 在发布文件中查找用于 Windows 的 UART 驱动程序，CP210x_VCP_Windows。
- 需要连接 USB0_UART0 到计算机。

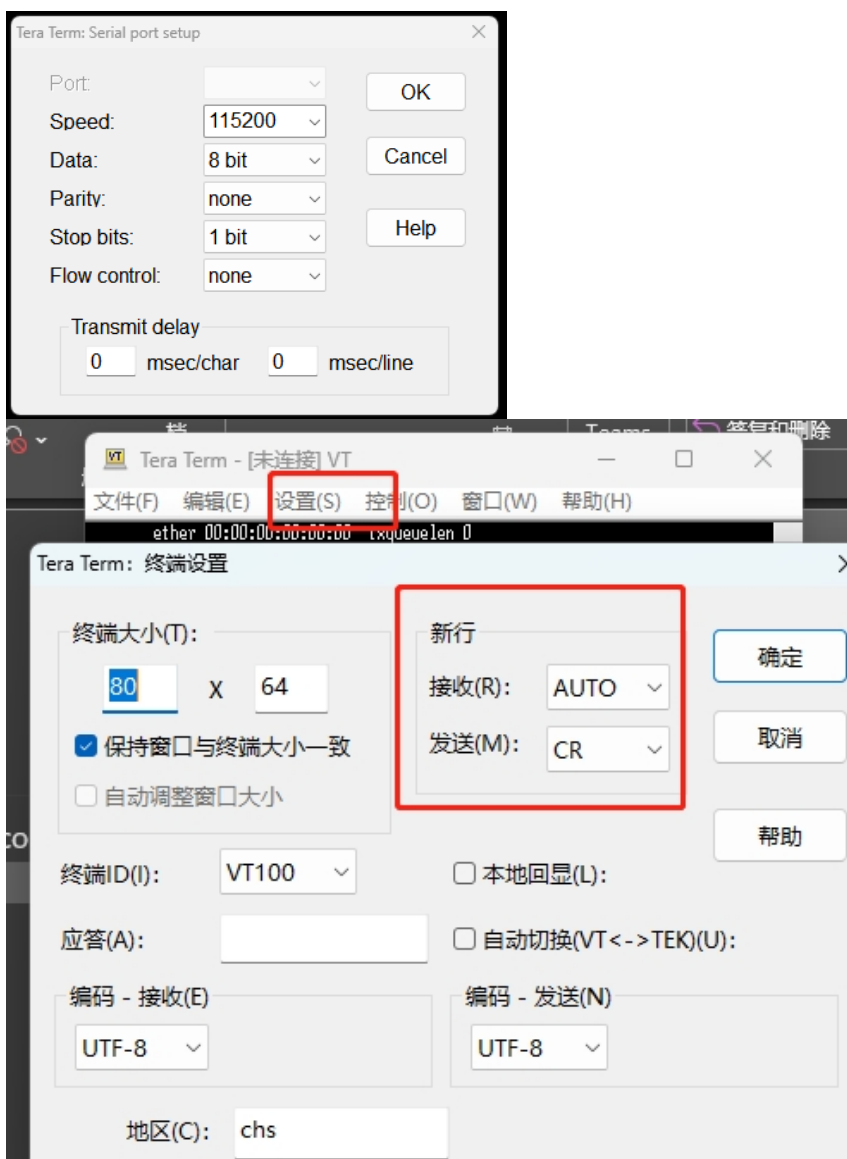


对于 UART 引导模式，SCM1612 将等待用户下载 RAM 可执行固件。

对于 XIP 引导模式，SCM1612 将在通电时从闪存引导。

对于 USB/SDIO 下载模式，一般用在 Host 模式下，通过 USB/SDIO 将可运行程序加载到 RAM 中并执行。

SCM1612 中有两个 UART 接口。对于 UART 引导模式，串口设置如下：



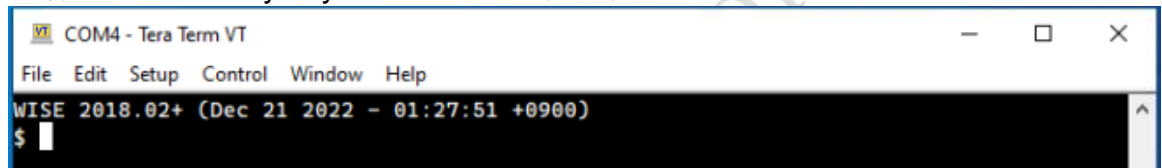
XIP bootloader 未烧录前，烧写 SCM1612 的一般步骤如下：

1. 设置 SW2 成 UART 引导模式，下载并运行 RAM 引导程序。
2. 烧录 XIP 引导程序。
3. 烧录 XIP 固件。
4. 设置 SW2 成 XIP 引导模式，复位并自动运行。

一旦 XIP 引导程序已烧录（如上述步骤 2），用户只需继续第 3 步进行开发。

5.1 在 RAM 上运行引导程序：

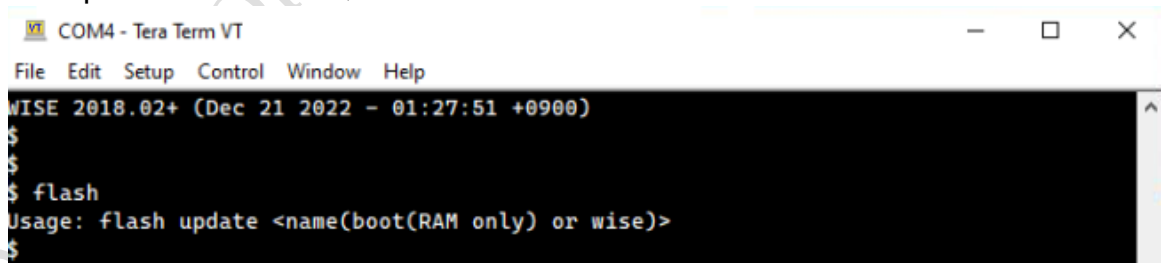
1. 设置 UART 引导模式并复位板子。
2. UART0 Teraterm 将周期性输出'C'，表示设备正在等待固件传输。
3. 发送“wise.scmboot.ram.bin”。
 - A. 从 Teraterm 菜单，[文件] -> [传输] -> [Ymodem] -> [发送] -> 从固件所在目录选择“wise.scmboot.ram.bin”。
 - B. 等待固件传输完成。
4. 固件传输成功后，将被执行。
5. 引导程序执行后，接下来将使用 UART0 Teraterm。
通过中断引导程序执行以进入引导程序 shell。
当看到“Press any key...3”消息时，按下任一键



6. 如果用户没有键入任何键，则引导程序将尝试执行主固件。如果发生这种情况，请从步骤 2) 重复。

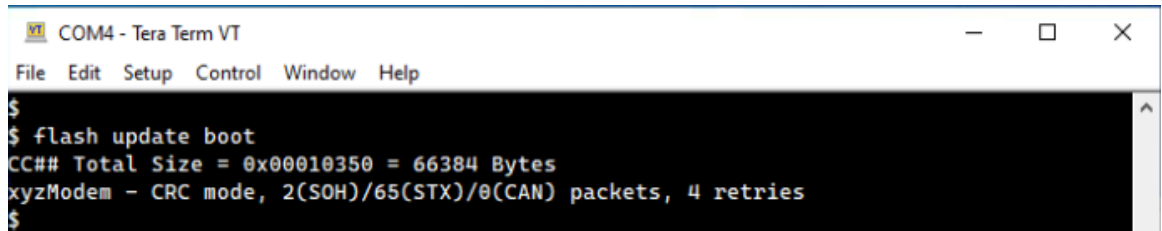


7. 当通过按键中断引导程序时，Wise shell 可用于进一步操作。尝试“help”或“flash”命令。



5.2 烧录 XIP 引导程序

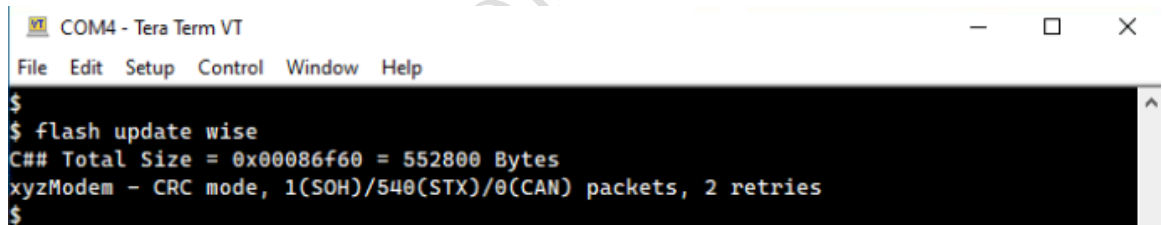
1. 输入以下命令：
`$ flash update boot`
2. 从 Teraterm 菜单，[文件] -> [传输] -> [Ymodem] -> [发送] -> 选择
“wise.scmboot.bin”。



```
COM4 - Tera Term VT
File Edit Setup Control Window Help
$
$ flash update boot
CC## Total Size = 0x00010350 = 66384 Bytes
xyzModem - CRC mode, 2(SOH)/65(STX)/0(CAN) packets, 4 retries
$
```

5.3 烧录主固件

1. 输入以下命令
`$ flash update wise`
2. 从 Teraterm 菜单，[文件] -> [传输] -> [Ymodem] -> [发送] -> 选择
“wise.mcuboot.bin”。



```
COM4 - Tera Term VT
File Edit Setup Control Window Help
$
$ flash update wise
C## Total Size = 0x00086f60 = 552800 Bytes
xyzModem - CRC mode, 1(SOH)/540(STX)/0(CAN) packets, 2 retries
$
```

5.4 自动运行固件

当所有固件文件烧录完成后，切换到 XIP 引导模式，然后复位板子。

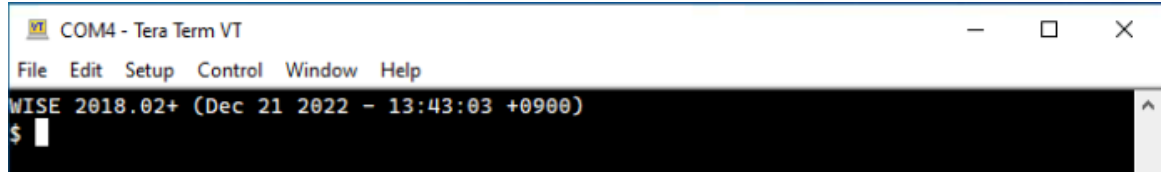
成功启动后，

- 1) UART0 首先显示引导程序日志



```
COM4 - Tera Term VT
File Edit Setup Control Window Help
Press any key...1
*** Booting MCUBoot ***
Booting from 0x80040000Hello world!
```

2) UART0 接下来显示主固件日志



5.5 仅更新固件

一旦 XIP 引导程序已烧录，可以使用 XIP 引导程序再次更新固件。XIP 引导程序提供与 RAM 引导程序相同的功能。

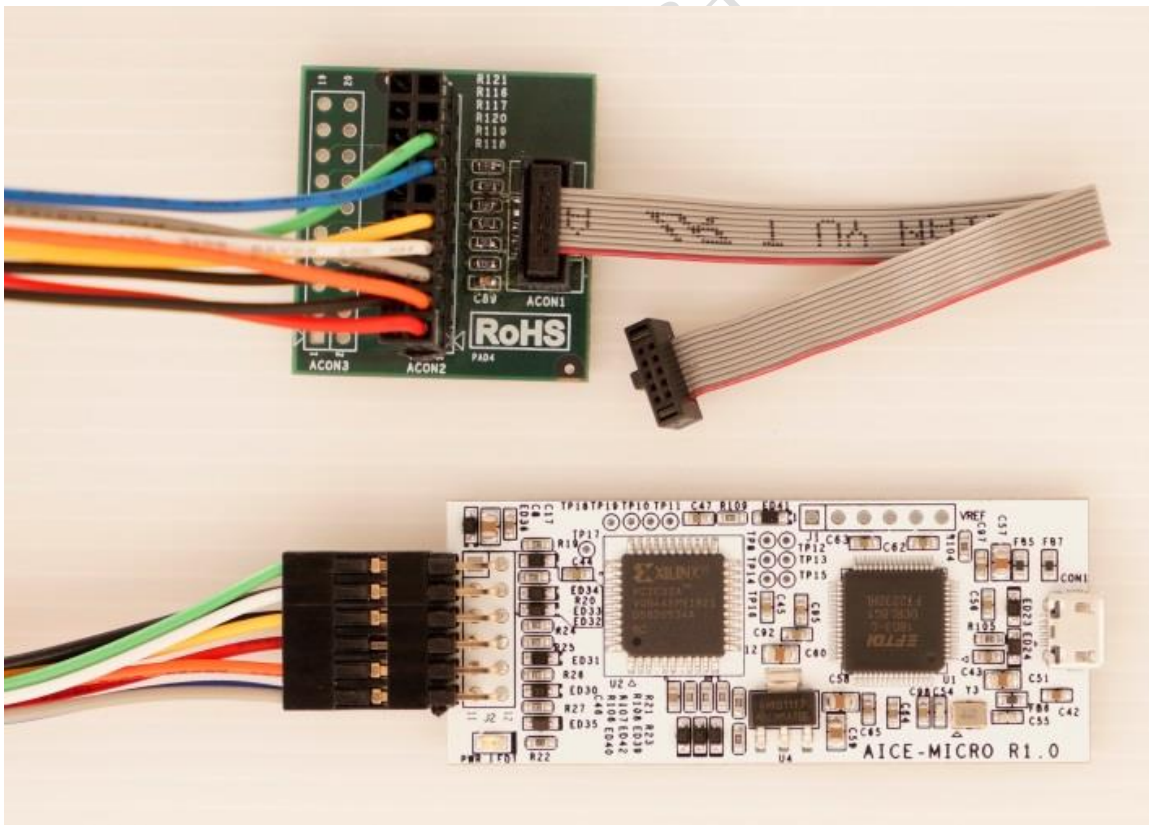
中断 XIP 引导程序加载并按照第 [5.3](#) 节的相同步骤进行操作。

6 调试

6.1 JTAG 调试使用 OpenOCD

6.1.1 概述

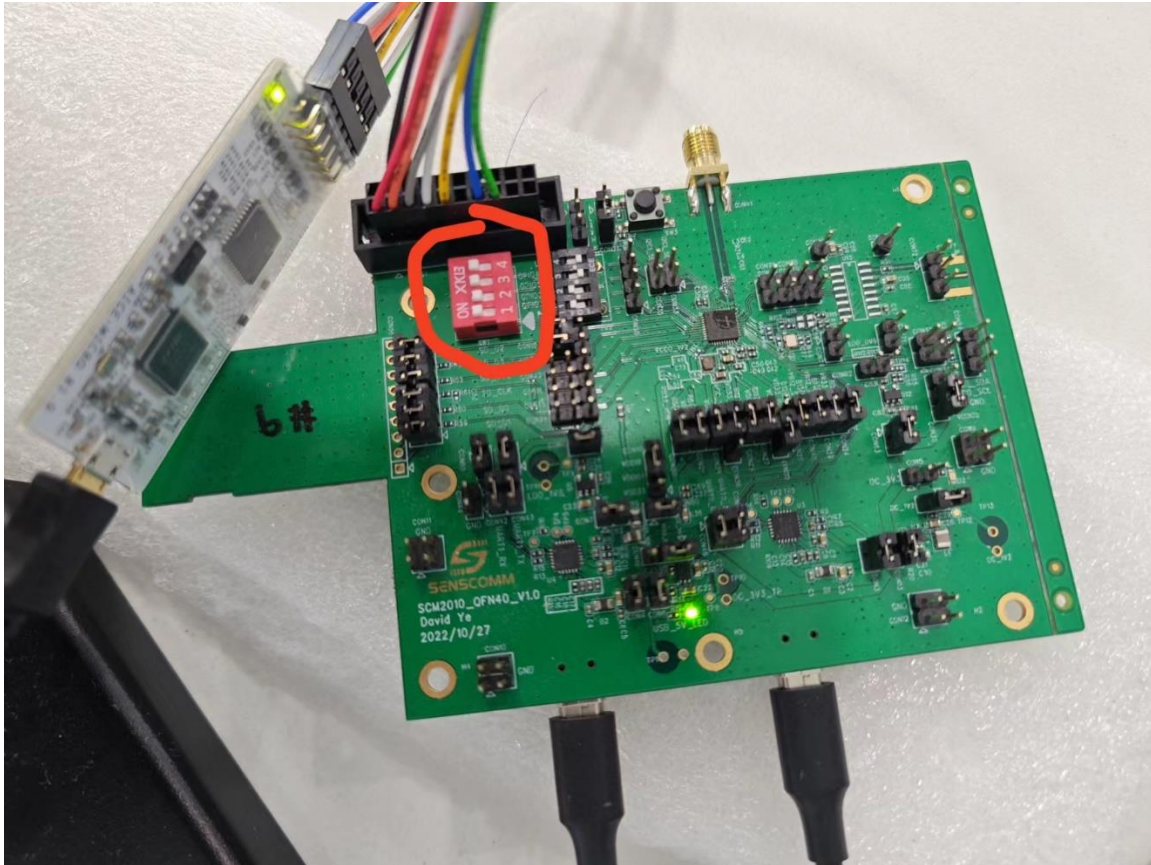
AndeShape AICE-MICRO 是一个基于 FT2232H 的 JTAG 调试设备，与 AndeSight™开发套件和 AndesCore V5 系列兼容，并支持 OpenOCD 的 JTAG 接口。



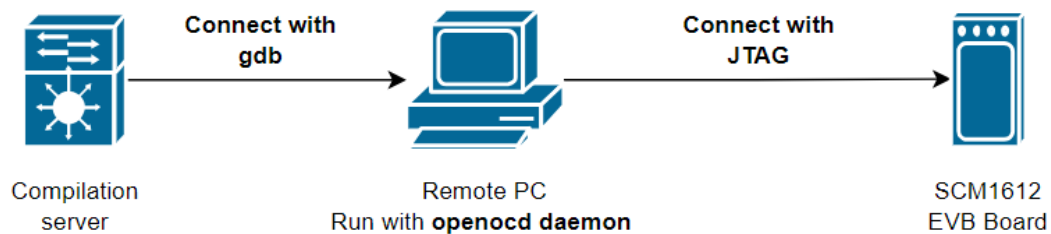
6.1.2 设置调试环境

将 JTAG 接口连接到 1612 EVB 板。请注意图像中红色圆圈区域。

在重启过程中，为完成 boot 操作，请确保 SW1 的 1-4 号拨码均处于下拉状态。待系统重新启动后，再将 1-4 号拨码拉起，以激活 JTAG 功能。



网络拓扑如下图所示。



6.1.3 远程调试

- a) 在远程计算机上运行 openocd.exe 守护程序。

```
C:\Work\openocd>openocd.exe
Open On-Chip Debugger 0.10.0+dev-ge990efd64-dirty (2022-06-13-15:31)
Licensed under GNU GPL v2
For bug reports, read
    http://openocd.org/doc/doxygen/bugs.html
JTAG frequency 10.000 MHz
The core #0 listens on 1234.
The core #1 listens on 1235.
ICEman is ready to use.
|
```

在编译服务器上，使用 `nds32le-elf-mculib-v5` 安装 `riscv32-elf-gdb`。

使用以下命令进行远程调试：

```
/opt/nds32le-elf-mculib-v5/bin/riscv32-elf-gdb
target remote 10.12.7.102:1234
```

说明：IP 地址 10.12.7.102 是用于连接 JTAG 的主机地址，而 1234 是目标端口。

```
GNU gdb (2022-02-07_riscv32-elf-0278d8cc40b) 8.2.50.20190322-git
Copyright (C) 2019 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "--host=x86_64-pc-linux-gnu --target=riscv32-elf".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word".
[info] Loading .Andesgdbinit.
[info] .Andesgdbinit loaded.
(gdb) target remote 10.12.7.102:1234
Remote debugging using 10.12.7.102:1234
warning: No executable has been specified and target does not support
determining executable automatically. Try using the "file" command.
0x00000056 in ?? ()
tap0_target_0(gdb) █
```