



StarFive  
赛昉科技

# 昉·星光 2 SDK快速参考手册

版本：1.24

日期：2024/4/3

Doc ID: VisionFive 2-QSGCH-002

# 法律声明

阅读本文件前的重要法律告知。

## 版权注释

版权 ©上海赛昉科技有限公司，2023。版权所有。

本文档中的说明均基于“视为正确”提供，可能包含部分错误。内容可能因产品开发而定期更新或修订。上海赛昉科技有限公司（以下简称“赛昉科技”）保留对本协议中的任何内容进行更改的权利，恕不另行通知。

赛昉科技明确否认任何形式的担保、解释和条件，无论是明示的还是默示的，包括但不限于适销性、特定用途适用性和非侵权的担保或条件。

赛昉科技无需承担因应用或使用任何产品或电路而产生的任何责任，并明确表示无需承担任何及所有连带责任，包括但不限于间接、偶然、特殊、惩戒性或由此造成的损害。

本文件中的所有材料受版权保护，为赛昉科技所有。不得以任何方式修改、编辑或断章取义本文件中的说明，本文件或其任何部分仅限用于内部使用或教育培训。

## 联系我们：

地址：浦东新区盛夏路61弄张润大厦2号楼502，上海市，201203，中国

网站：<http://www.starfivetech.com>

邮箱：

- [sales@starfivetech.com](mailto:sales@starfivetech.com)（销售）
- [support@starfivetech.com](mailto:support@starfivetech.com)（支持）

# 目录

表格清单.....	4
插图清单.....	5
法律声明.....	ii
前言.....	vi
<b>1. 简介.....</b>	<b>8</b>
<b>2. 前期准备.....</b>	<b>9</b>
<b>3. 下载SDK.....</b>	<b>10</b>
<b>4. 构建指令.....</b>	<b>11</b>
<b>5. 构建Buildroot、U-Boot、Linux内核和BusyBox.....</b>	<b>12</b>
<b>6. 在昉·星光 2上运行SDK.....</b>	<b>16</b>
6.1. 将昉·星光 2连接到网络.....	16
6.2. 启动昉·星光 2.....	18
6.2.1. 使用默认的DTB文件运行image.fit.....	19
6.2.2. 使用其他DTB文件运行Image.gz和initramfs.cpio.gz.....	20
<b>7. 附录.....</b>	<b>22</b>
7.1. 创建TF卡镜像.....	22
7.2. 动态加载DTB Overlay.....	24
7.3. 更新Flash中的SPL和U-Boot.....	24
7.4. 恢复Bootloader.....	28
7.5. 昉·星光 2启动模式设置.....	33

# 表格清单

表 0-1 修订历史.....	vi
表 7-1 启动模式设置.....	33



# 插图清单

图 5-1 示例输出.....	12
图 5-2 示例输出.....	13
图 5-3 示例输出.....	13
图 5-4 示例输出.....	14
图 5-5 示例输出.....	14
图 5-6 示例输出.....	15
图 6-1 连接昉·星光 240-Pin GPIO Header的Debug Pin.....	16
图 7-1 连接昉·星光 240-Pin GPIO Header的Debug Pin.....	28
图 7-2 启动模式设置(UART).....	29
图 7-3 示例输出.....	29
图 7-5 示例输出.....	31
图 7-7 示例输出.....	32
图 7-9 启动模式设置位置.....	34
图 7-10 启动模式设置.....	35

# 前言

关于本指南和技术支持信息

## 关于本手册

本手册为SDK开发者配置昉·星光 2单板计算机提供快速参考指南。昉·星光 2是全球首款集成了GPU的高性能RISC-V单板计算机，搭载赛昉科技新一代SoC平台——昉·惊鸿-7110。



### 重要：

赛昉科技为开发者提供了两种文档类型：网页版和PDF版。在执行命令时，为避免格式有误，请在网页版文档中复制命令。






## 修订历史

表 0-1 修订历史

版本	发布说明	修订
1.24	2024/04/03	<ul style="list-style-type: none"><li>在<a href="#">构建Buildroot、U-Boot、Linux内核和BusyBox (第 12页)</a>新增了示例输出图片。</li><li>在<a href="#">前言 (第 vi页)</a>新增了一个注释。</li></ul>
1.23	2024/01/30	更新了 <a href="#">启动昉·星光 2 (第 18页)</a> 一节。
1.22	2023/11/24	更新了 <a href="#">前期准备 (第 9页)</a> 中安装依赖包的命令。
1.21	2023/05/10	更新了 <a href="#">昉·星光 2启动模式设置 (第 33页)</a> 。
1.2	2023/4/19	<ul style="list-style-type: none"><li>更新了<a href="#">恢复Bootloader (第 28页)</a>中恢复文件的版本信息；</li><li>在<a href="#">更新Flash中的SPL和U-Boot (第 24页)</a>中添加了一个注释。</li></ul>
1.1	2022/12/27	<ul style="list-style-type: none"><li>在<a href="#">更新Flash中的SPL和U-Boot (第 24页)</a>新增了一个更新方法。</li><li>在<a href="#">恢复Bootloader (第 28页)</a>增加了示例输出图片。</li></ul>
1.0	2022/12/15	首次发布。

## 注释和注意事项

本指南中可能会出现以下注释和注意事项：

-  **提示：**  
建议如何在某个主题或步骤中应用信息。
-  **注：**  
解释某个特例或阐释一个重要的点。
-  **重要：**  
指出与某个主题或步骤有关的重要信息。
-  **警告：**  
表明某个操作或步骤可能会导致数据丢失、安全问题或性能问题。
-  **警告：**  
表明某个操作或步骤可能导致物理伤害或硬件损坏。

---

# 1. 简介

赛昉科技提供昉·星光 2 软件开发工具包 (SDK) 来构建U-Boot SPL和U-Boot, 并提供带有OpenSBI二进制、Linux内核、设备树、ramdisk和rootdisk的扁平镜像树 (FIT) 镜像, 同时也为昉·星光 2构建了完整的RISC-V交叉编译 toolchain。

按照以下步骤使用SDK:

1. [前期准备 \(第 9页\)](#)
2. [下载SDK \(第 10页\)](#)
3. [构建指令 \(第 11页\)](#)
4. [构建Buildroot、U-Boot、Linux内核和BusyBox \(第 12页\)](#)
5. [在昉·星光 2上运行SDK \(第 16页\)](#)



## 2. 前期准备

在使用SDK前，请确保您已执行以下步骤：

1. 在您的个人电脑上安装操作系统。



**提示：**

推荐安装的操作系統为Ubuntu 16.04/18.04/ 20.04。

2. 执行以下命令，以更新所有软件包：

```
$sudo apt update  
$sudo apt upgrade
```

3. 执行以下命令，以安装所需的其他软件包：

```
$ sudo apt-get install build-essential automake libtool texinfo bison  
flex gawk g++ git xxd curl wget gdisk gperf cpio bc screen texinfo  
unzip libgmp-dev libmpfr-dev libmpc-dev libssl-dev libncurses-dev  
libglib2.0-dev libpixmap-1-dev libyaml-dev patchutils python3-pip  
zlib1g-dev device-tree-compiler dosfstools mtools kpartx rsync
```



**重要：**

该步骤的安装命令仅适用于VF2\_v3.4.5及之后的SDK版本。若您的SDK为VF2\_v3.1.5或之前的版本，请使用以下命令安装所需的其他软件包：

```
$ sudo apt-get install build-essential g++ git autoconf  
automake autotools-dev texinfo bison xxd curl flex gawk gdisk  
gperf libgmp-dev libmpfr-dev libmpc-dev libz-dev libssl-dev  
libncurses-dev libtool patchutils python screen texinfo  
unzip zlib1g-dev libyaml-dev wget cpio bc dosfstools mtools  
device-tree-compiler libglib2.0-dev libpixmap-1-dev kpartx
```

4. 执行以下命令，为Git LFS安装其他软件包：

```
$ curl -s  
https://packagecloud.io/install/repositories/github/git-lfs/script.deb  
.sh | sudo bash  
$ sudo apt-get install git-lfs
```

## 3. 下载SDK

按照以下步骤，从赛昉科技官方GitHub代码仓下载SDK：

1. 执行以下命令，从SDK代码仓下载代码（例如，`JH7110_VisionFive2_devel`），并下载所有相关子模块：

```
$ git clone git@github.com:starfive-tech/VisionFive2.git
$ cd VisionFive2
$ git checkout JH7110_VisionFive2_devel
$ git submodule update --init --recursive
```



### 注：

下载过程需要一些时间，并将占用大约5 GB的磁盘空间。由于部分依赖模块托管在Git上，下载可能会失败。如果遇到此类情况，赛昉科技建议您稍后再试，或请已经完成下载的人员提供副本拷贝。

2. （可选）对于构建发布tag版的用户，只需执行第一步。对于需要切换4个子模块（即`buildroot`、`u-boot`、`linux`和`opensbi`）到正确分支的开发人员，执行以下命令，可以参考`.gitmodule`文件以获得正确的分支信息。

```
$ cd buildroot && git checkout --track origin/<buildroot_branch> &&
cd ..
$ cd u-boot && git checkout --track origin/<u-boot_branch> && cd ..
$ cd linux && git checkout --track origin/<linux_branch> && cd ..
$ cd opensbi && git checkout <opensbi_branch> && cd ..
```

以下为示例命令：

```
$ cd buildroot && git checkout --track origin/JH7110_VisionFive2_devel
&& cd ..
$ cd u-boot && git checkout --track origin/JH7110_VisionFive2_devel &&
cd ..
$ cd linux && git checkout --track origin/JH7110_VisionFive2_devel &&
cd ..
$ cd opensbi && git checkout master && cd ..
```

## 4. 构建指令

本节提供在更新了[下载SDK \(第 10页\)](#)子模块后构建指令的步骤。这个过程是对initramfs镜像image.fit的快速构建，它可以通过TFTP转到开发板上，并在开发板上运行。

1. 执行以下命令，以创建toolchain, u-boot-spl.bin.normal.out, visionfive2\_fw\_payload.img和image.fit。

```
make -j$(nproc)
```

### 结果:

将在work/目录下生成以下文件:

```
work/
├─ visionfive2_fw_payload.img
├─ image.fit
├─ initramfs.cpio.gz
├─ u-boot-spl.bin.normal.out
├─ linux/arch/riscv/boot
│   └─ dts
│       └─ starfive
│           ├── jh7110-visionfive-v2-ac108.dtb
│           ├── jh7110-visionfive-v2.dtb
│           ├── jh7110-visionfive-v2-wm8960.dtb
│           ├── vf2-overlay
│           └─ vf2-overlay-uart3-i2c.dtbo
└─ Image.gz
```

最终的build tree将占用大约16 GB的磁盘空间。

2. 将此前生成的文件复制到TFTP服务器的工作区路径下。

# 5. 构建Buildroot、U-Boot、Linux内核和BusyBox

使用以下命令构建Buildroot、U-Boot、Linux内核和BusyBox。

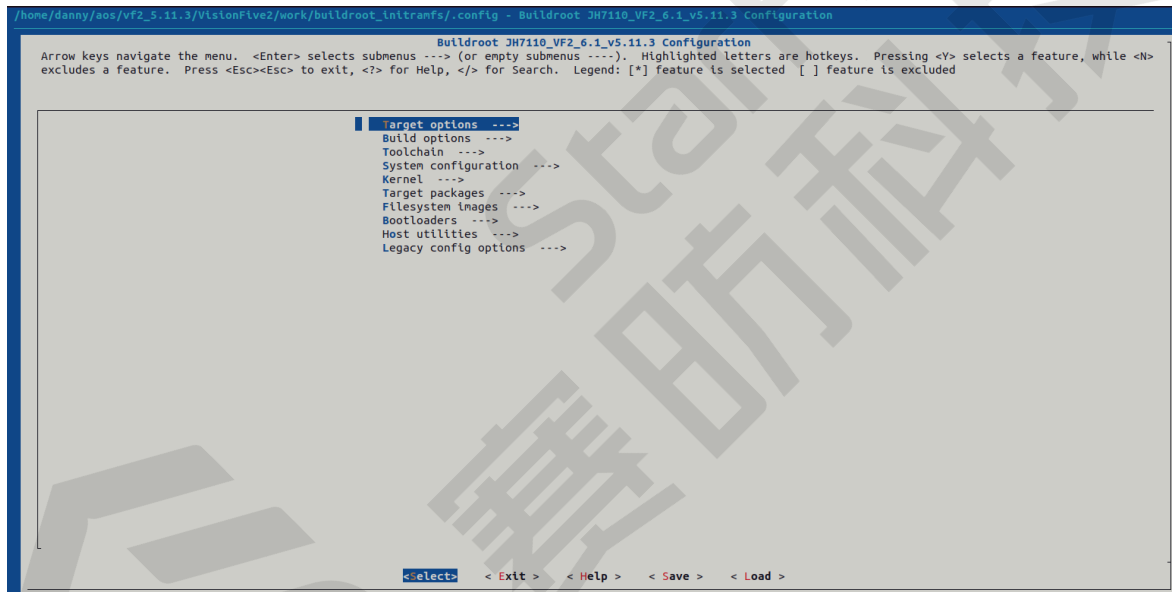
## 构建Buildroot

Buildroot是简单、高效和易用的工具，通过交叉编译可生成嵌入式Linux系统。

执行以下命令，在您的开发板上构建Buildroot：

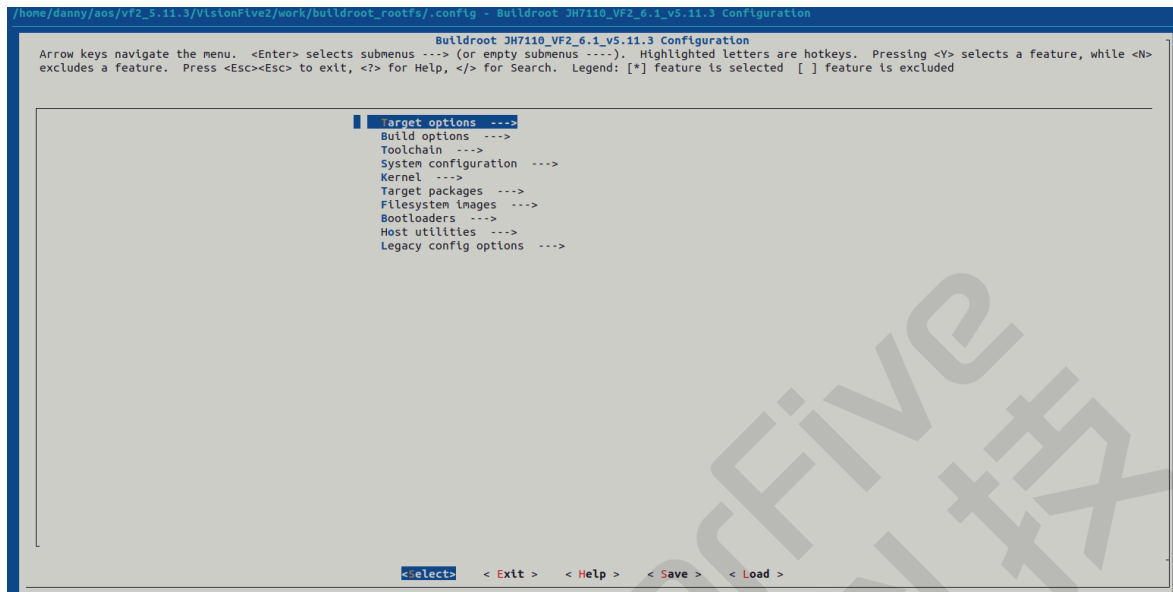
- `$ make buildroot_initramfs-menuconfig # initramfs menuconfig`

图 5-1 示例输出



- `$ make buildroot_rootfs-menuconfig # rootfs menuconfig`

图 5-2 示例输出



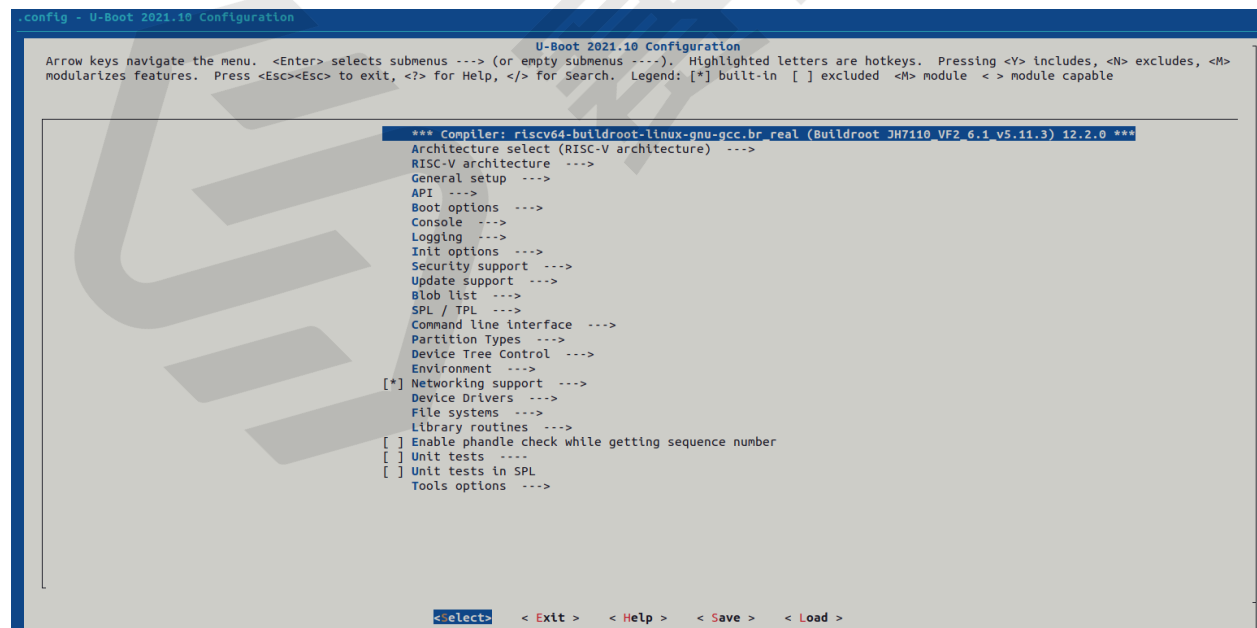
## 构建U-Boot:

*Universal Boot Loader (U-Boot)*是一个开源的、用于嵌入式系统的引导加载程序。

执行以下命令，在您的开发板上构建U-Boot:

```
$ make uboot-menuconfig # uboot menuconfig
```

图 5-3 示例输出



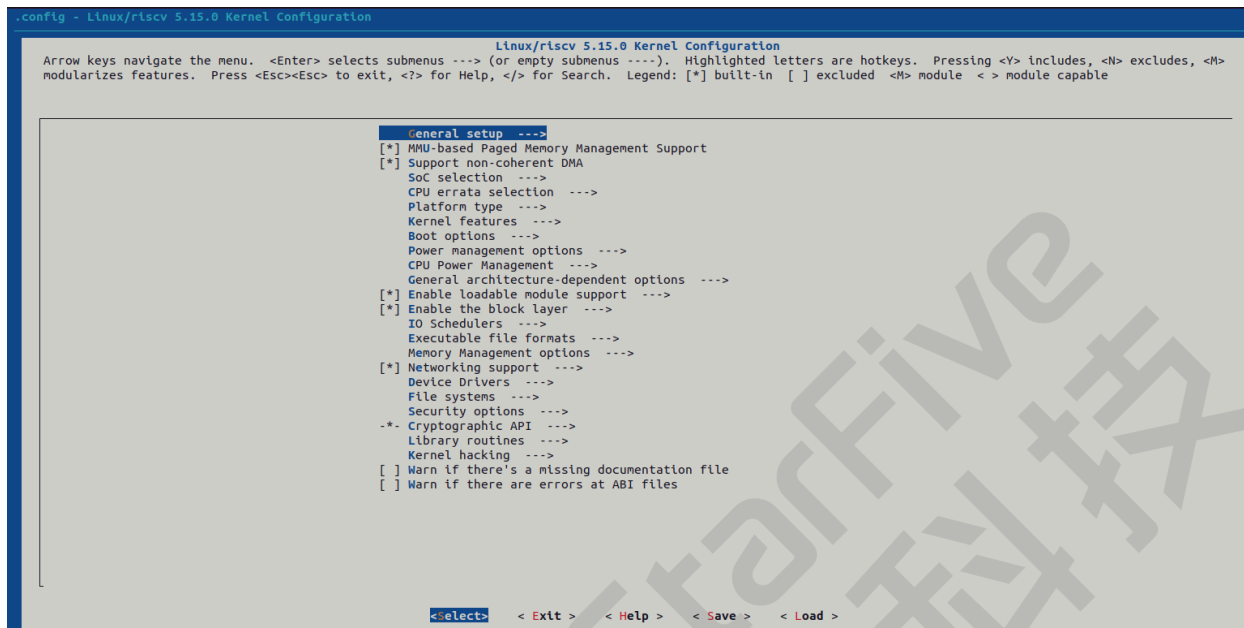
## 构建Linux内核

Linux内核是Linux操作系统的主要组件，是计算机硬件与其进程之间的核心接口。

执行以下命令，在您的开发板上构建Linux内核：

```
$ make linux-menuconfig # Kernel menuconfig
```

图 5-4 示例输出



## 构建BusyBox

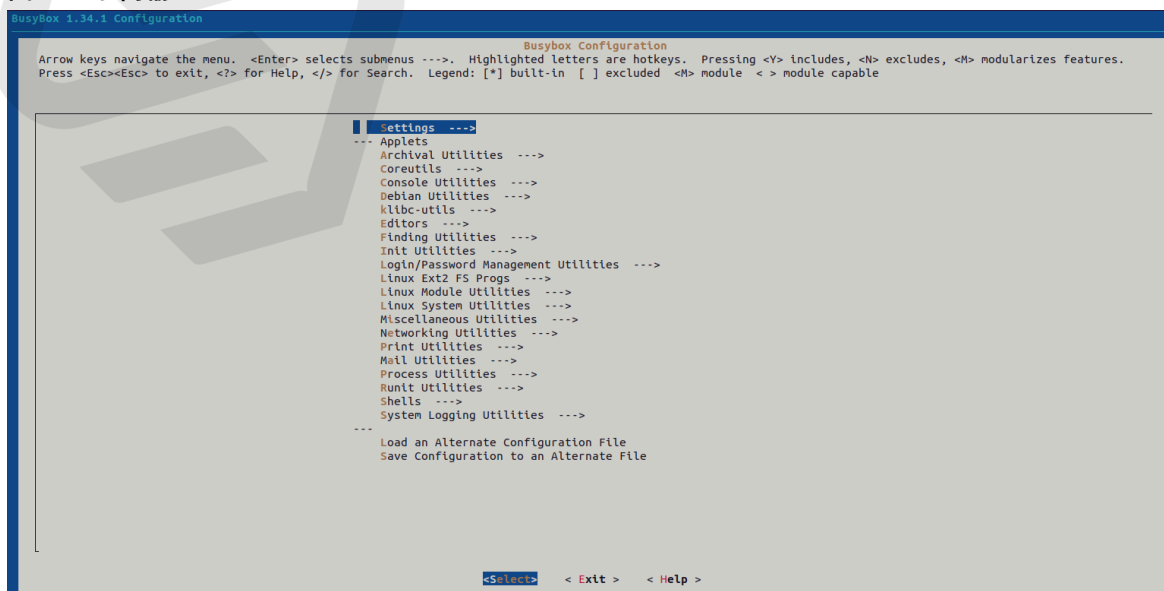
BusyBox是一套简单的工具集，包含许多精简的Linux实用程序。

执行以下命令，在您的开发板上构建BusyBox菜单配置：

- 使用Initramfs文件系统为BusyBox构建菜单配置：

```
$ make -C ./work/buildroot_initramfs/ O=./work/buildroot_initramfs
busybox-menuconfig
```

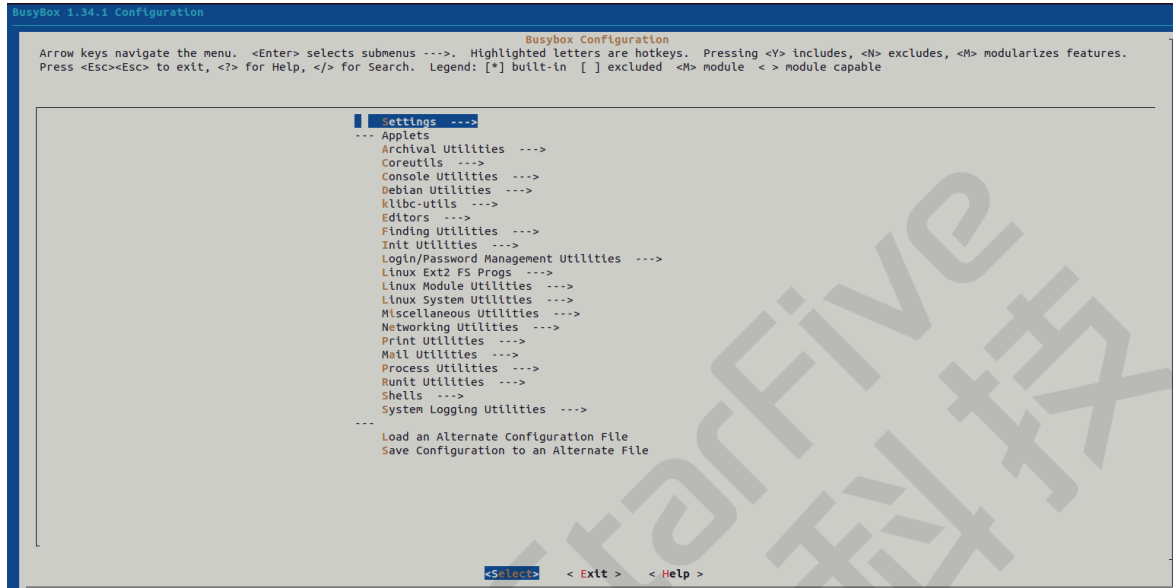
图 5-5 示例输出



- 使用Rootfs文件系统为BusyBox构建菜单配置：

```
$ make -C ./work/buildroot_rootfs/ O=./work/buildroot_rootfs
busybox-menuconfig
```

图 5-6 示例输出



## 构建Linux内核、BusyBox和FFmpeg

如果您想构建单个包或模块，根据您的需求选择执行下面的命令：

- 构建Linux内核：

```
$ make vmlinux
```

- 构建BusyBox：

```
make -C ./work/buildroot_rootfs/ O=./work/buildroot_rootfs
busybox-rebuild # build busybox package
```

- 构建FFmpeg包：

```
$ make -C ./work/buildroot_rootfs/ O=./work/buildroot_rootfs #
ffmpeg-rebuild
```

## 6. 在昉·星光 2上运行SDK

按照以下步骤，在昉·星光 2上运行SDK：

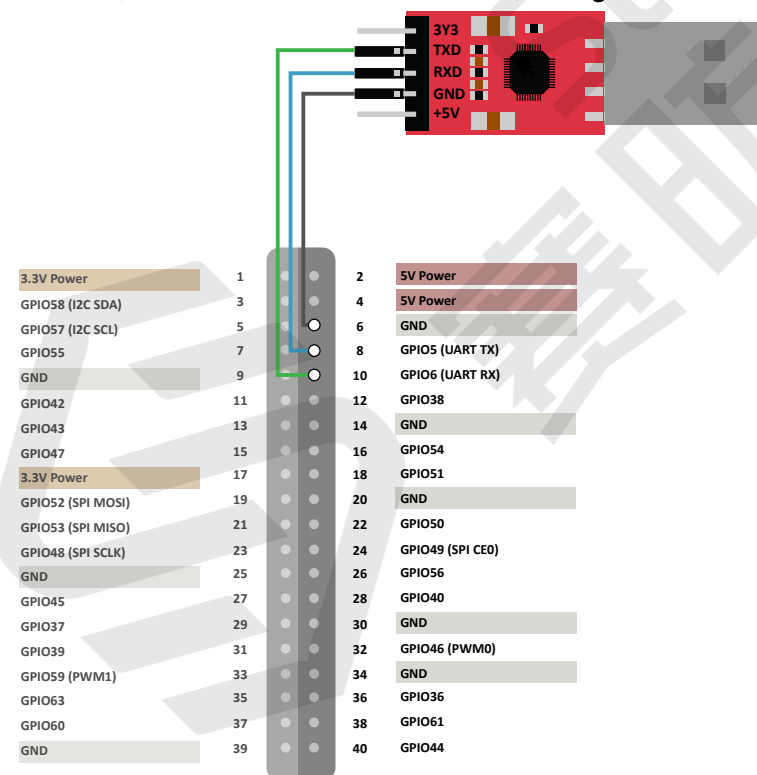
1. 如[将昉·星光 2连接到网络 \(第 16页\)](#)所述，将昉·星光 2连接到网络。
2. 选择以下方式之一启动昉·星光 2：
  - [使用默认的DTB文件运行image.fit \(第 19页\)](#)
  - [使用其他DTB文件运行Image.gz和initramfs.cpio.gz \(第 20页\)](#)

### 6.1. 将昉·星光 2连接到网络

本节提供将昉·星光 2连接到网络和进入U-Boot终端的步骤。

1. 将USB转串口转换器的跳线连接到昉·星光 240-Pin GPIO Header的Debug pin上。下图为示例：

图 6-1 连接昉·星光 240-Pin GPIO Header的Debug Pin



2. 将串口波特率设置为115200 bps。
3. 将昉·星光 2接上网线和电源线。
4. 启动昉·星光 2，您将看到以下启动信息：



```

U-Boot SPL 2021.10 (Oct 31 2022 - 12:11:37 +0800)
DDR version: dc2e84f0.
Trying to boot from SPI
OpenSBI v1.0

  _____
 /  _  \ /  ___|  _  \  _
|  |  |  _  _  _  _  |  (  _  |  |  )  |  |
|  |  |  '  _  \ /  _  \ '  _  \  \  _  \ |  _  <  |  |
|  |  _  |  |  |  )  |  _  /  |  |  |  _  )  |  |  )  |  |  _
 \  _  / |  .  _  /  \  _  |  _  |  |  _  /  |  _  /  _  /  _  |
  |  |
  |  |
  |  |
Platform Name : StarFive VisionFive V2
Platform Features : medeleg
Platform HART Count : 5
Platform IPI Device : aclint-mswi
Platform Timer Device : aclint-mtimer @ 4000000Hz
Platform Console Device : uart8250
Platform HSM Device : ---
Platform Reboot Device : ---
Platform Shutdown Device : ---
Firmware Base : 0x40000000
Firmware Size : 360 KB
Runtime SBI Version : 0.3
Domain0 Name : root
Domain0 Boot HART : 3
Domain0 HARTs : 0*,1*,2*,3*,4*
Domain0 Region00 : 0x0000000002000000-0x000000000200ffff (I)
Domain0 Region01 : 0x0000000040000000-0x000000004007ffff ( )
Domain0 Region02 : 0x0000000000000000-0xfffffffffffffffffff
(R,W,X)
Domain0 Next Address : 0x0000000040200000
Domain0 Next Arg1 : 0x0000000042200000
Domain0 Next Mode : S-mode
Domain0 SysReset : yes
Boot HART ID : 3
Boot HART Domain : root
Boot HART Priv Version : v1.11
Boot HART Base ISA : rv64imafdcbx
Boot HART ISA Extensions : none
Boot HART PMP Count : 8
Boot HART PMP Granularity : 4096
Boot HART PMP Address Bits: 34
Boot HART MHPM Count : 2
Boot HART MIDELEG : 0x0000000000000222
Boot HART MEDELEG : 0x000000000000b109
U-Boot 2021.10 (Oct 31 2022 - 12:11:37 +0800), Build:
jenkins-VF2_515_Branch_SDK_Release-10
CPU: rv64imacu

```

```
Model: StarFive VisionFive V2
DRAM: 8 GiB
MMC: sdio0@16010000: 0, sdio1@16020000: 1
Loading Environment from SPIFlash... SF: Detected gd25lq128 with page
size 256 Bytes, erase size 4 KiB, total 16 MiB
*** Warning - bad CRC, using default environment
StarFive EEPROM format v2
-----EEPROM INFO-----
Vendor : StarFive Technology Co., Ltd.
Product full SN: VF7110A1-2243-D008E000-00000001
data version: 0x2
PCB revision: 0xa1
BOM revision: A
Ethernet MAC0 address: 6c:cf:39:00:14:5b
Ethernet MAC1 address: 6c:cf:39:00:14:5c
-----EEPROM INFO-----
In: serial@10000000
Out: serial@10000000
Err: serial@10000000
Model: StarFive VisionFive V2
Net: eth0: ethernet@16030000, eth1: ethernet@16040000
switch to partitions #0, OK
mmc1 is current device
found device 1
bootmode flash device 1
Failed to load 'uEnv.txt'
Can't set block device
Hit any key to stop autoboot: 0
StarFive #
```

5. 按任意键停止，进入U-Boot终端。

## 6.2. 启动昉·星光 2

昉·星光 2 的启动方式有两种，您可以选择以下两种中的一种。

- [使用默认的DTB文件运行image.fit \(第 19页\)](#)
- [使用其他DTB文件运行Image.gz和initramfs.cpio.gz \(第 20页\)](#)

不同的开发板使用不同的dtb文件：

- `jh7110-visionfive-v2.dtb`：用于1.2A版的开发板。
- `jh7110-visionfive-v2-ac108.dtb`：用于带有ac108编解码器的1.2A版的开发板。
- `jh7110-visionfive-v2-wm8960.dtb`：用于带有wm8960编解码器的1.2A版的开发板。

**提示：**

您可查看开发板上的丝印获取版本信息。

## 6.2.1. 使用默认的DTB文件运行image.fit

本节提供通过TFTP传输image.fit并使用默认的DTB文件：jh7110-visionfivev2.dtb运行image.fit的步骤。

1. 运行以下命令，设置环境参数：

```
setenv ipaddr 192.168.xxx.xxx; setenv serverip 192.168.xxx.xxx;
```

2. 将镜像文件上传至DDR：

```
tftpboot ${loadaddr} image.fit;
```

3. 运行以下命令，下载并执行文件：

```
bootm start ${loadaddr};bootm loados ${loadaddr};run
chipa_set_linux;run cpu_vol_set; booti ${kernel_addr_r}
${ramdisk_addr_r}:${filesize} ${fdt_addr_r};
```

4. 使用以下凭据登录：

```
buildroot login:root
Password: starfive
```

**结果：**

启动成功！

**重要：**

以上命令仅适用于VF2\_v3.4.5及之后的SDK版本。若您的SDK为VF2\_v3.1.5或之前的版本，请使用以下命令运行image.fit：

- a. 运行以下命令，设置环境参数：

```
setenv bootfile vmlinuz; setenv fdtcontroladdr
0xffffffffffffffff; setenv fileaddr a0000000; setenv
ipaddr 192.168.xxx.xxx; setenv serverip 192.168.xxx.xxx;
```

- b. 将镜像文件上传至DDR：

```
tftpboot ${fileaddr} ${serverip}:image.fit;
```

- c. 运行以下命令，下载并执行文件：



```
bootm start ${fileaddr};bootm loados ${fileaddr};run
chipa_set_linux;booti 0x40200000 0x46100000:${filesize}
0x46000000
```

d. 使用以下凭据登录:

```
buildroot login:root
Password: starfive
```

**结果:**

启动成功!

## 6.2.2. 使用其他DTB文件运行Image.gz和initramfs.cpio.gz

如果您想加载其他DTB文件, 例如jh7110-visionfive-v2-wm8960.dtb, 请按照以下操作步骤加载:

1. 设置环境参数:

```
setenv ipaddr 192.168.xxx.xxx; setenv serverip 192.168.xxx.xxx;
```

2. 上传文件至DDR:

```
tftpboot ${fdt_addr_r} jh7110-visionfive-v2-wm8960.dtb;
tftpboot ${kernel_addr_r} Image.gz;
tftpboot ${ramdisk_addr_r} initramfs.cpio.gz;
run chipa_set_linux;run cpu_vol_set;
```

3. 加载并执行该文件:

```
booti ${kernel_addr_r} ${ramdisk_addr_r}:${filesize} ${fdt_addr_r}
```

4. 使用以下凭据登录:

```
buildroot login:root
Password: starfive
```

**结果:**

启动成功。



**重要:**

以上命令仅适用于VF2\_v3.4.5及之后的SDK版本。若您的SDK为VF2\_v3.1.5或之前的版本, 请使用以下命令运行Image.gz和initramfs.cpio.gz:



a. 运行以下命令，设置环境参数：

```
setenv bootfile vmlinuz; setenv
fdtcontroladdr 0xffffffffffffffff; setenv fileaddr
a0000000; setenv ipaddr 192.168.xxx.xxx; setenv
serverip 192.168.xxx.xxx;
setenv kernel_comp_addr_r 0xb0000000;setenv
kernel_comp_size 0x10000000;
```

b. 将镜像文件上传至DDR：

```
tftpboot ${fdt_addr_r} jh7110-visionfive-v2-wm8960.dtb;
tftpboot ${kernel_addr_r} Image.gz;
tftpboot ${ramdisk_addr_r} initramfs.cpio.gz;
run chipa_set_linux;
```

c. 运行以下命令，下载并执行文件：

```
booti ${kernel_addr_r} ${ramdisk_addr_r}:${filesize}
${fdt_addr_r}
```

d. 使用以下凭据登录：

```
buildroot login:root
Password: starfive
```

## 7. 附录

### 7.1. 创建TF卡镜像

如果您没有部署本地TFTP服务器，则您可以使用TF卡进行数据传输。



**重要：**

创建TF卡镜像将覆盖目标TF卡上的所有现有数据。

TF卡的默认大小为16 GB，建议您使用TF卡的GPT分区表。

按照以下步骤，创建TF卡镜像：

1. 执行以下命令，生成sdcard.img文件。

```
$ make -j$(nproc)
$ make buildroot_rootfs -j$(nproc)
$ make img
```

**结果：**

将生成输出文件work/sdcard.img。



**提示：**

您可以通过以下方式之一将镜像文件烧录到TF卡上：

- 执行以下dd命令：

```
$ sudo dd if=work/sdcard.img of=/dev/sdX bs=4096
$ sync
```

- 或使用rpi-imager或balenaEtcher工具。

2. (可选) 如有需要，可使用以下方法扩展分区。

- 方法1：在Ubuntu主机上：

- a. 输入以下命令，安装软件包：

```
$ sudo apt install cloud-guest-utils e2fsprogs
```

- b. 在Ubuntu主机上插入一张TF卡。

- c. 执行以下命令扩展分区。



**注：**

/dev/sdX是TF卡设备名，可根据实际情况更改该变量。

```

$ sudo growpart /dev/sdX 4 # extend partition 4
$ sudo e2fsck -f /dev/sdX4
$ sudo resize2fs /dev/sdX4 # extend filesystem
$ sudo fsck.ext4 /dev/sdX4

```

- 方法2: 在昉·星光 2开发板上运行fdisk和resize2fs命令:

```

# fdisk /dev/mmcblk1
Welcome to fdisk (util-linux 2.37.2).
Changes will remain in memory only, until you decide to write
them.
Be careful before using the write command.
This disk is currently in use - repartitioning is probably a bad
idea.
It's recommended to umount all file systems, and swapoff all swap
partitions on this disk.
Command (m for help): d
Partition number (1-4, default 4): 4
Partition 4 has been deleted.
Command (m for help): n
Partition number (4-128, default 4): 4
First sector (614400-62333918, default 614400):
): t sector, +/-sectors or +/-size{K,M,G,T,P} (614400-62333918,
default 62333918)
Created a new partition 4 of type 'Linux filesystem' and of size
29.4 GiB.
Partition #4 contains a ext4 signature.
Do you want to remove the signature? [Y]es/[N]o: N
Command (m for help): w
The partition table has been altered.
Syncing disks.
# resize2fs /dev/mmcblk1p4
resize2fs 1.46.4 (18-Aug-2021)
Filesystem at /d[
111.756178] EXT4-fs (mmcblk1p4): resizing filesystem from 512000
to 30859756 blocks
ev/mmcblk1p4 is [
111.765203] EXT4-fs (mmcblk1p4): resizing filesystem from 512000
to 30859265 blocks
mounted on /; on-line resizing required
old_desc_blocks = 2, new_desc_blocks = 118
[ 112.141953] random: crng init done
[ 112.145369] random: 7 urandom warning(s) missed due to
ratelimiting
[ 115.474184] EXT4-fs (mmcblk1p4): resized filesystem to 30859265
The filesystem on /dev/mmcblk1p4 is now 30859756 (1k) blocks long.

```

## 7.2. 动态加载DTB Overlay

系统支持在开发板运行时动态加载DTB overlay。

```
mount -t configfs none /sys/kernel/config
mkdir -p /sys/kernel/config/device-tree/overlays/dtoverlay
cd <the dtoverlay.dtbo path>
cat vf2-overlay-uart3-i2c.dtbo
> /sys/kernel/config/device-tree/overlays/dtoverlay/dtbo
```

另外，您可以执行以下命令，删除DTBO（设备树覆盖）功能：

```
rmdir /sys/kernel/config/device-tree/overlays/dtoverlay
```

## 7.3. 更新Flash中的SPL和U-Boot

以下提供更新昉·星光 2 Flash中的SPL和U-Boot的两种方法：



**注：**

如需获取创建SPL和fw\_payload（U-Boot）文件的参考手册，请参阅[《昉·星光 2单板计算机软件技术参考手册》](#)中的“创建SPL文件”和“创建fw\_payload文件”。

1. 通过tftpboot命令更新SPL和U-Boot。
2. 通过flashcp命令更新SPL和U-Boot。



**注：**

方法2仅支持镜像版本为VF2\_v2.5.0或高于该版本的镜像。

### 通过tftpboot命令

通过tftpboot命令更新SPL和U-Boot，请执行以下步骤：



**注：**

1-7步是在主机PC上进行，8-13步是在昉·星光 2上进行。

1. 将以太网电缆的一端连接到昉·星光 2 RJ45接口上，将另一端连接到路由器上。
2. 在主机PC上安装TFTP服务器：

```
sudo apt-get update
sudo apt install tftpd-hpa
```

3. 检查服务器状态：



```
sudo systemctl status tftpd-hpa
```

4. 输入以下命令进入TFTP服务器:

```
sudo nano /etc/default/tftpd-hpa
```

5. 执行以下命令设置TFTP服务器:

```
TFTP_USERNAME="tftp"
TFTP_DIRECTORY="/home/user/Desktop/tftp_share"
TFTP_ADDRESS=":69"
TFTP_OPTIONS="--secure"
```



**注:**

TFTP\_DIRECTORY指bootloader、u-boot、SPL和镜像等文件的存放路径。

6. 执行以下命令，创建tftp-server文件夹以存储文件:

```
sudo mkdir -p /home/user/tftp
```

7. 重启TFTP服务器:

```
sudo systemctl restart tftpd-hpa
```

8. 打开昉·星光 2，等待它进入U-Boot命令行界面。



**提示:**

- 前提条件:
  - 一个USB转TTL转换器，将USB连接到电脑，杜邦线连接到昉·星光 2正确的扩展引脚上，特别注意TX和RX需要交叉对接
  - 在您的电脑上安装Putty或者secureCRT等。
- 当您给昉·星光 2上电后，串口会打印数字并且进行递减计时，一般是从3开始递减，在数字减为0之前点击任意按键就可以进入U-Boot的命令行模式。

9. 执行以下命令设置环境变量:

```
setenv ipaddr 192.168.120.222;setenv serverip 192.168.120.99
```



**注:**

一般情况下路由器的默认IP为 192.168.120.1。在这种情况下，请使用由路由器的DHCP服务器分配的IP，昉·星光 2的IP地址应为192.168.120.xxx。但是，如果



您的路由器IP不同（例如192.168.2.1），请确保服务器IP和昉·星光 2属于同一IP段（例如192.168.2.xxx）中。

10. 使用ping命令，检查主机与昉·星光 2的连接情况。

**示例命令：**

```
ping 192.168.120.99
```

**结果：**

以下输出表明主机与昉·星光单板计算机已经在同一网络下建立连接。

```
StarFive # ping 192.168.120.99
speed: 1000, full duplex
Using dwmac.10020000 device
host 192.168.120.99 is alive
```

11. 初始化SPI flash:

```
sf probe
```

**结果：**

```
StarFive # sf probe
SF: Detected gd25lq128 with page size 256 Bytes, erase size 4 KiB,
total 16 MiB
```

12. 更新SPL二进制文件，以下为命令和示例输出：

```
StarFive # tftpboot 0xa0000000 ${serverip}:u-boot-spl.bin.normal.out
Using ethernet@16030000 device
TFTP from server 192.168.120.99; our IP address is 192.168.120.222
Filename 'u-boot-spl.bin.normal.out'.
Load address: 0xa0000000
Loading: #####
1.6 MiB/s
done
Bytes transferred = 132208 (20470 hex)
StarFive # sf update 0xa0000000 0x0 $filesize
device 0 offset 0x0, size 0x20470
0 bytes written, 132208 bytes skipped in 0.23s, speed 5206961 B/s
```

13. 更新U-Boot二进制文件，以下为命令和示例输出：

```
StarFive # tftpboot 0xa0000000 ${serverip}:visionfive2_fw_payload.img
Using ethernet@16030000 device
TFTP from server 192.168.120.99; our IP address is 192.168.120.222
Filename 'visionfive2_fw_payload.img'.
Load address: 0xa0000000
Loading:
```

```
#####
#####
#####
#####
2.2 MiB/s
done
Bytes transferred = 2955397 (2d1885 hex)
StarFive # sf update 0xa0000000 0x100000 $filesize
device 0 offset 0x100000, size 0x2d1885
0 bytes written, 2955397 bytes skipped in 0.507s, speed 5922361 B/s
StarFive #
```

## 通过flashcp命令

通过flashcp命令更新SPL和U-Boot, 请执行以下步骤:



**注:**

方法2仅支持镜像版本为VF2\_v2.5.0或高于该版本的镜像。

1. 执行以下命令, 安装mtd-utils安装包:

```
apt install mtd-utils
```

2. 通过SCP将最新的u-boot-spl.bin.normal.out和visionfive2\_fw\_payload.img文件移植到Debian系统上。
3. 执行以下命令, 查看MTD分区:

```
cat /proc/mtd
```

### 示例输出:

您可以看到QSPI Flash里的数据分区:

```
dev: size erasesize name
mtd0: 00040000 00001000 "spl"
mtd1: 00010000 00001000 "uboot-env"
mtd2: 00300000 00001000 "uboot"
mtd3: 00100000 00001000 "data"
```

4. 根据不同分区的内容, 分别通过flashcp更新SPL和U-Boot:

- 更新SPL的示例命令:

```
flashcp -v u-boot-spl.bin.normal.out /dev/mtd0
```

- 更新U-Boot的示例命令:

```
flashcp -v visionfive2_fw_payload.img /dev/mtd2
```

## 示例命令和输出：

```
# flashcp -v u-boot-spl.bin.normal.out /dev/mtd0
Erasing blocks: 36/36 (100%)
Writing data: 143k/143k (100%)
Verifying data: 143k/143k (100%)

# flashcp -v visionfive2_fw_payload.img /dev/mtd2
Erasing blocks: 736/736 (100%)
Writing data: 2943k/2943k (100%)
Verifying data: 2943k/2943k (100%)
```

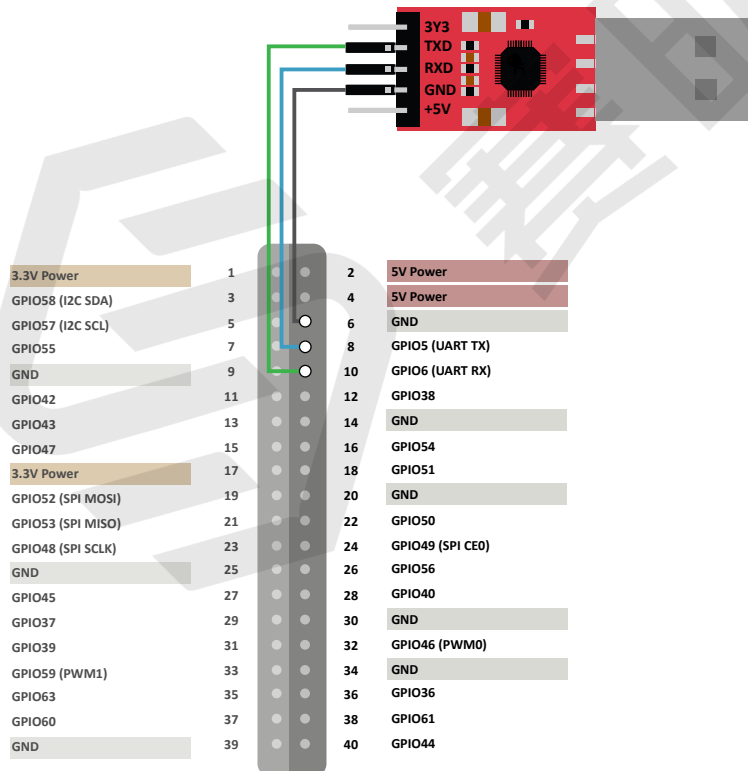
5. 重启系统，以使更新生效。

## 7.4. 恢复Bootloader

SPL和U-Boot储存在SPI flash中。您可能会意外清空闪存，或者昉·星光2的闪存损坏。在这些情况下，我们需要重置Bootloader。

1. 将USB转串口转换器的跳线连接到昉·星光 240-Pin GPIO Header的Debug pin上。下图为示例：

图 7-1 连接昉·星光 240-Pin GPIO Header的Debug Pin

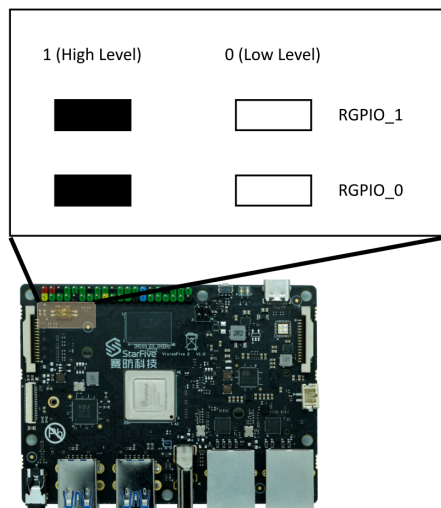


2. 在您恢复bootloader前，请再次检查昉·星光2上的启动模式跳线（Switch\_2）已经切换为UART模式（RGPIO\_1, RGPIO\_0: 1,1）。

**提示：**

下图为启动模式设置。更多信息请参见[昉·星光 2 启动模式设置 \(第 33页\)](#)。

图 7-2 启动模式设置(UART)



3. 将串口波特率设置为115200 bps。

4. 上电后，输出如下：

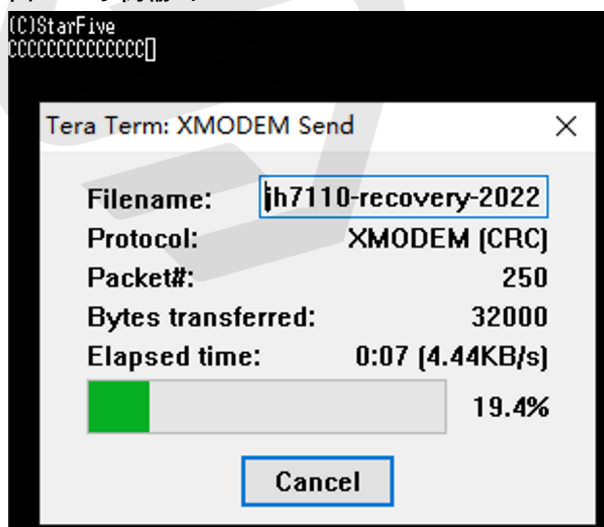
```
cccccccccccccccccccc
```

5. 通过XMODEM，传输恢复二进制文件（jh7110-recovery-<Version>.bin）。恢复二进制文件地址为：<https://github.com/starfive-tech/Tools/tree/master/recovery>。

**提示：**

<Version>表示二进制文件的版本号。请确保您使用的是最新的版本。

图 7-3 示例输出



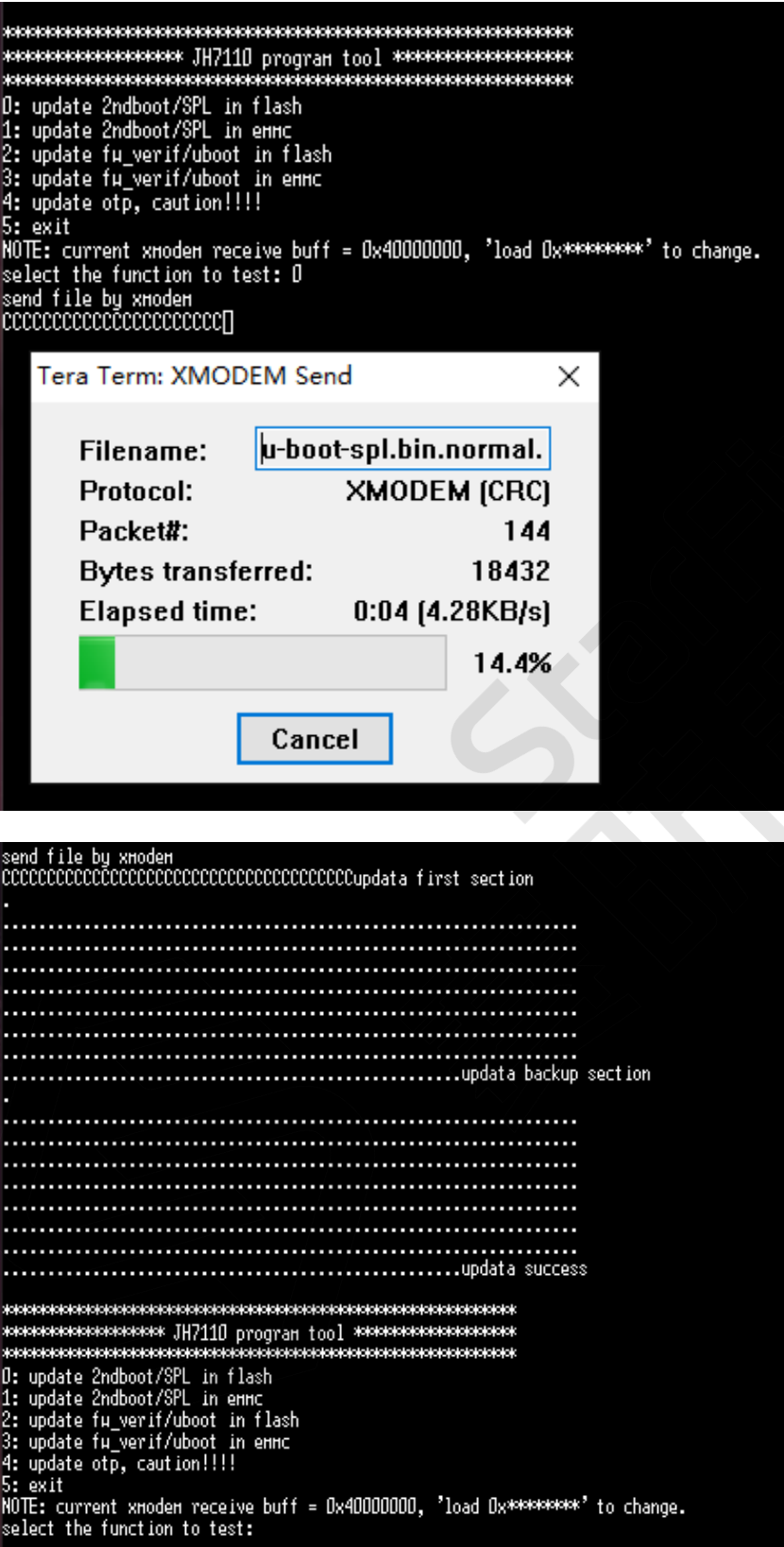
```
(C)StarFive
CCCCCCCCCCCC
JH7110 secondboot version: 221205-74596a9
CPU freq: 1250MHz
idcode: 0x1860C8
CSD:0xd00f0032 0x8f5903ff 0xffffffff 0x8a404023
mmc_send_ext_csd err 0
Device: EMMC
Manufacturer ID: 45
OEM: 100
Name: DG403
Tran Speed: 25000000
Rd Block Len: 512
MMC version 4.0
High Capacity: Yes
Capacity: 29.1 GiB
Bus Width: 8-bit
Erase Group Size: 0x80000
ddr 0x00000000, 4M test
ddr 0x00400000, 8M test
DDR clk 2133M, size 8GB

*****
***** JH7110 program tool *****
*****

0: update 2ndboot/SPL in flash
1: update 2ndboot/SPL in emmc
2: update fu_verif/u-boot in flash
3: update fu_verif/u-boot in emmc
4: update otp, caution!!!
5: exit
NOTE: current xmodem receive buff = 0x40000000, 'load 0x*****' to change.
select the function to test:
```

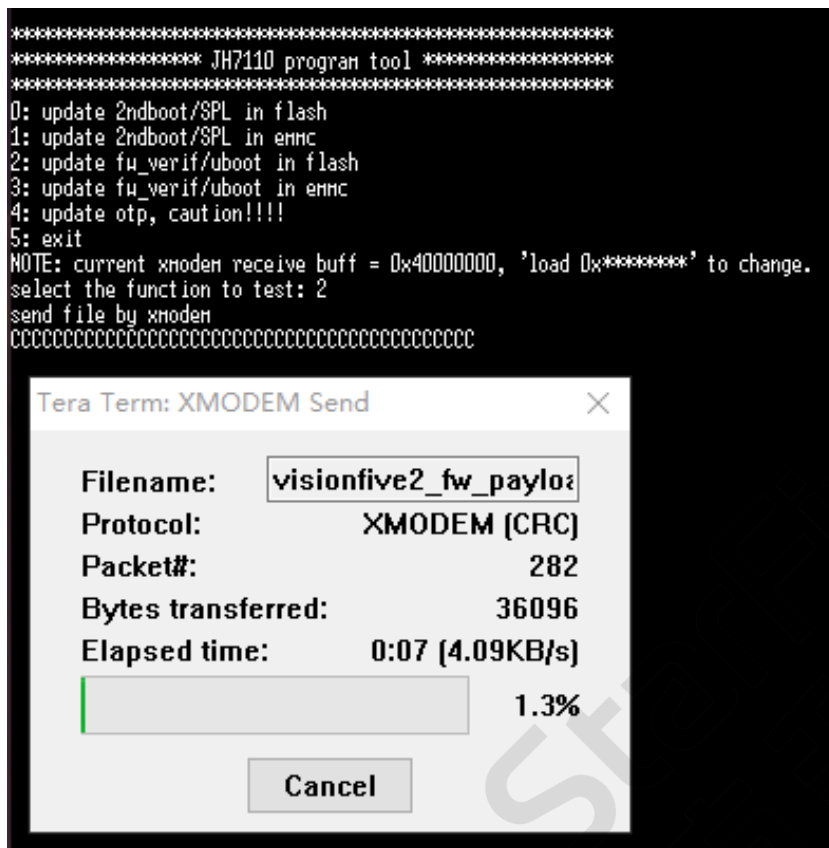
6. 输入0，按Enter键更新SPL二进制文件<u-boot-spl.bin.normal.out>。

图 7-5 示例输出



7. 输入2，按**Enter**键更新U-Boot二进制文件<visionfive2\_fw\_payload.img>。

图 7-7 示例输出





```

.....update success

*****
***** JH7110 program tool *****
*****

0: update 2ndboot/SPL in flash
1: update 2ndboot/SPL in emmc
2: update fu_verif/u-boot in flash
3: update fu_verif/u-boot in emmc
4: update otp, caution!!!
5: exit
NOTE: current xmodem receive buff = 0x40000000, 'load 0x*****' to change.
select the function to test:

```

8. 关闭电源并将跳线切换回Flash模式（RGPIO\_1, RGPIO\_0: 0, 0）。

## 7.5. 昉·星光 2 启动模式设置

昉·星光 2 提供专门的pin，帮助用户在上电前配置启动模式。以下是可选的启动模式及其详细信息。

表 7-1 启动模式设置

index	启动模式	RGPIO_1	RGPIO_0
1	1-bit QSPI Nor Flash	0 (L)	0 (L)
2	SDIO3.0	0 (L)	1 (H)
3	eMMC	1 (H)	0 (L)
4	UART	1 (H)	1 (H)



注：

赛昉科技建议您使用 *1-bit QSPI Nor Flash* 模式启动，因为使用 eMMC 或 SDIO3.0 启动模式可能会发生小概率启动失败的情况。如果从 eMMC 或 SDIO3.0 启动失败，您可以尝试重启昉·星光 2。

下图显示了启动模式专用 pin 的位置及其定义。

图 7-9 启动模式设置位置

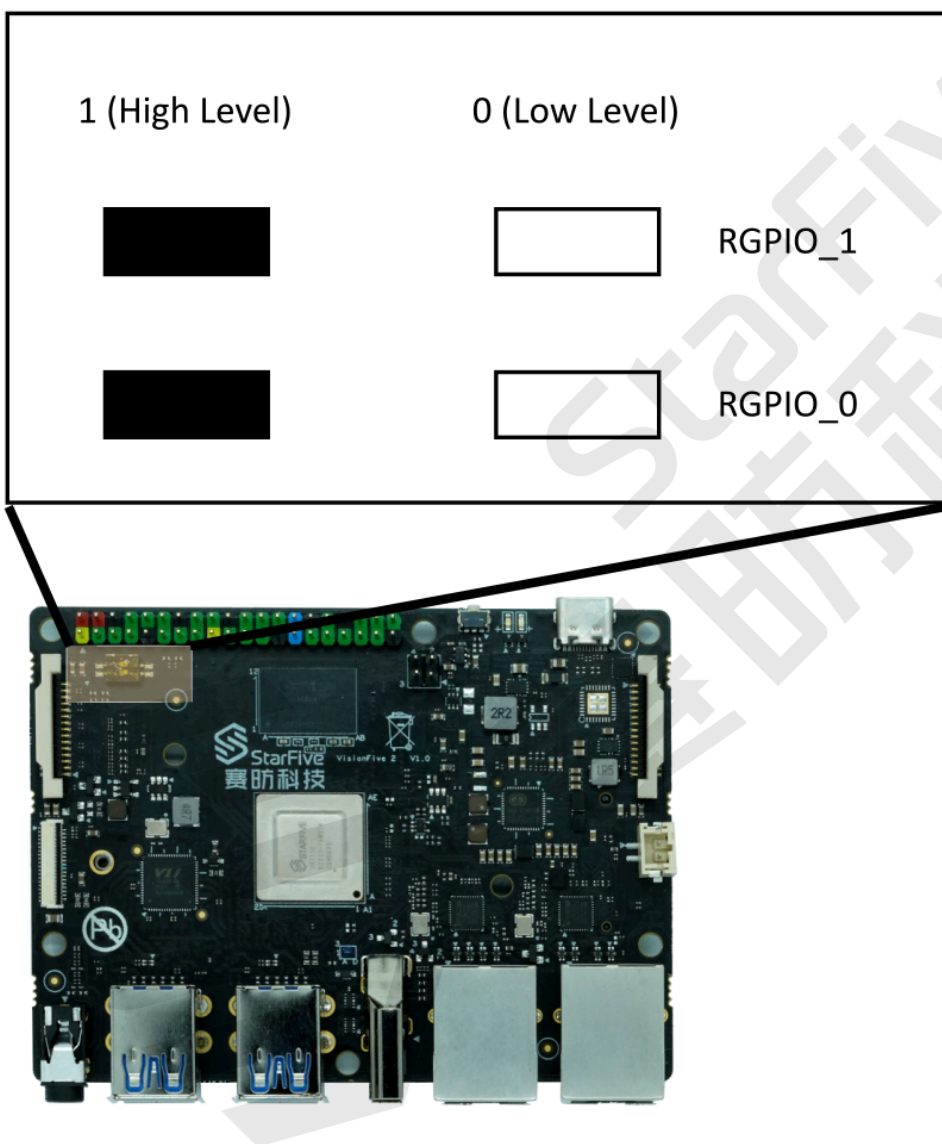


图 7-10 启动模式设置



**QSPI**  
 RGPI0\_1: 0 (L)  
 RGPI0\_0: 0 (L)



**SDIO**  
 RGPI0\_1: 0 (L)  
 RGPI0\_0: 1 (H)



**eMMC**  
 RGPI0\_1: 1 (H)  
 RGPI0\_0: 0 (L)



**UART**  
 RGPI0\_1: 1 (H)  
 RGPI0\_0: 1 (H)

Note: H for high level; L for low level.



注：  
 开发板版本不同，丝印可能也不同。