SigmaStar Camera
开发环境搭建

REVISION HISTORY

Revision No.	Description	Date
1.0	• Initial release	19/12/2018
1.1	• Add spi nand note.	22/01/2019

TABLE OF CONTENTS

1. 板子硬件连接

1.1. SSC009A 主板示意图:

电源： DC 12V,
调试串口： TTL 电平，特率 115200

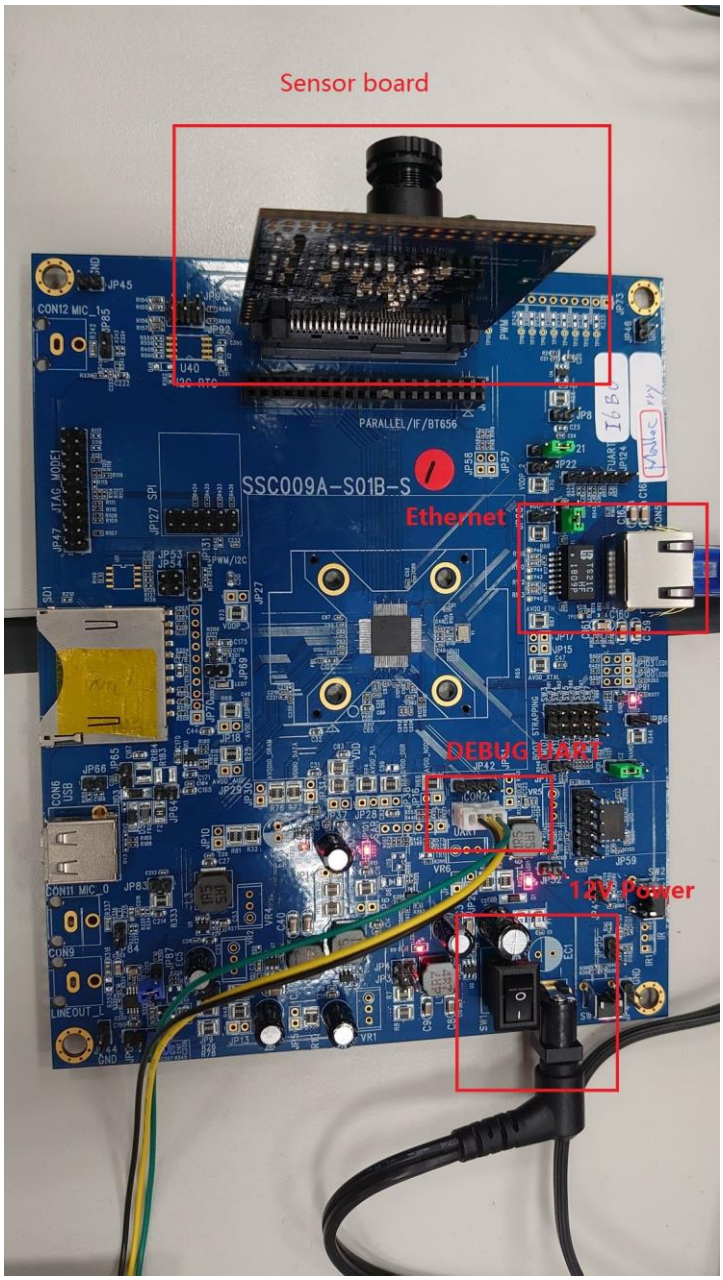

Figure 1: 图 1-1

SPI-NOR 启动的 jumper 配置：

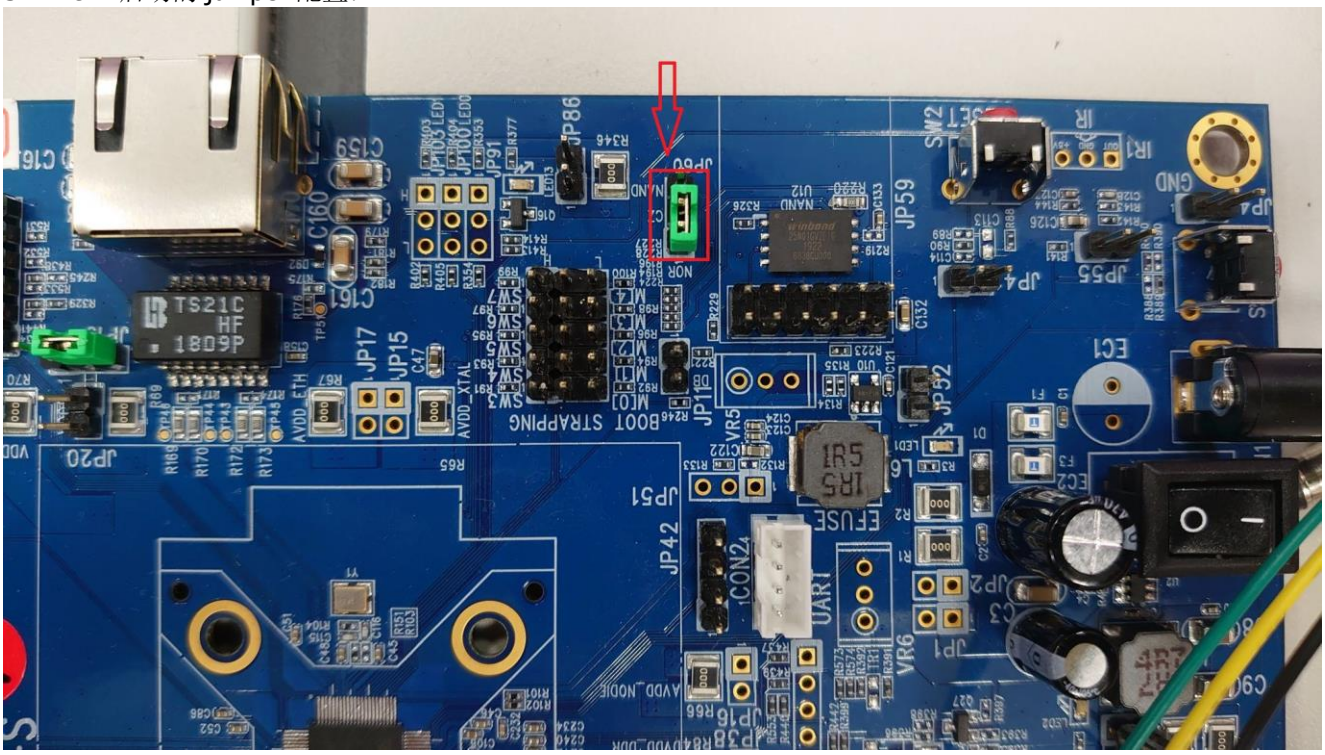

Figure 2 图 1-2

SPI-NAND 启动的 Jumper 配置：

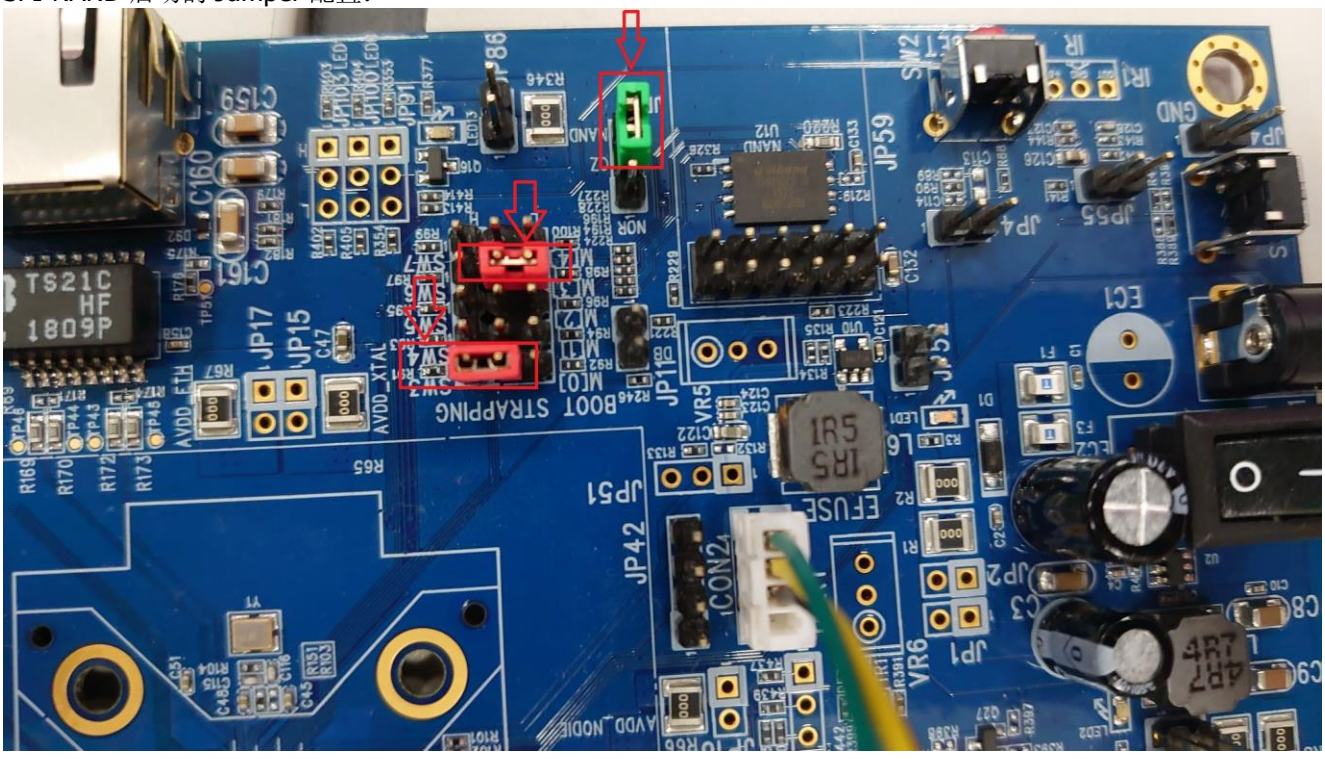

Figure 3 图 1-3

1.2. SSC009B 主板示意图：

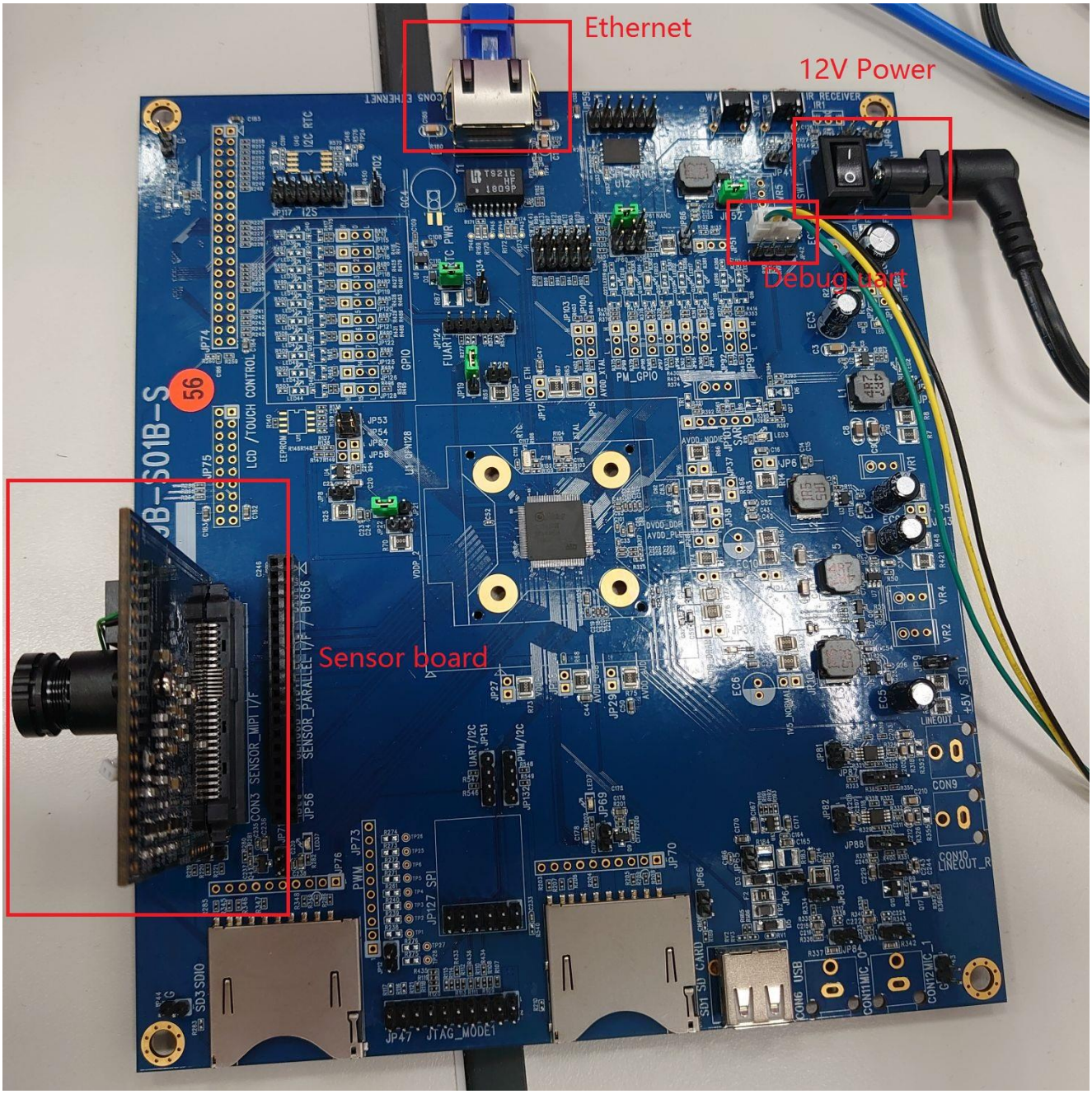

Figure 4: 图 1-4

SPI-NOR 启动的 Jumper 配置：

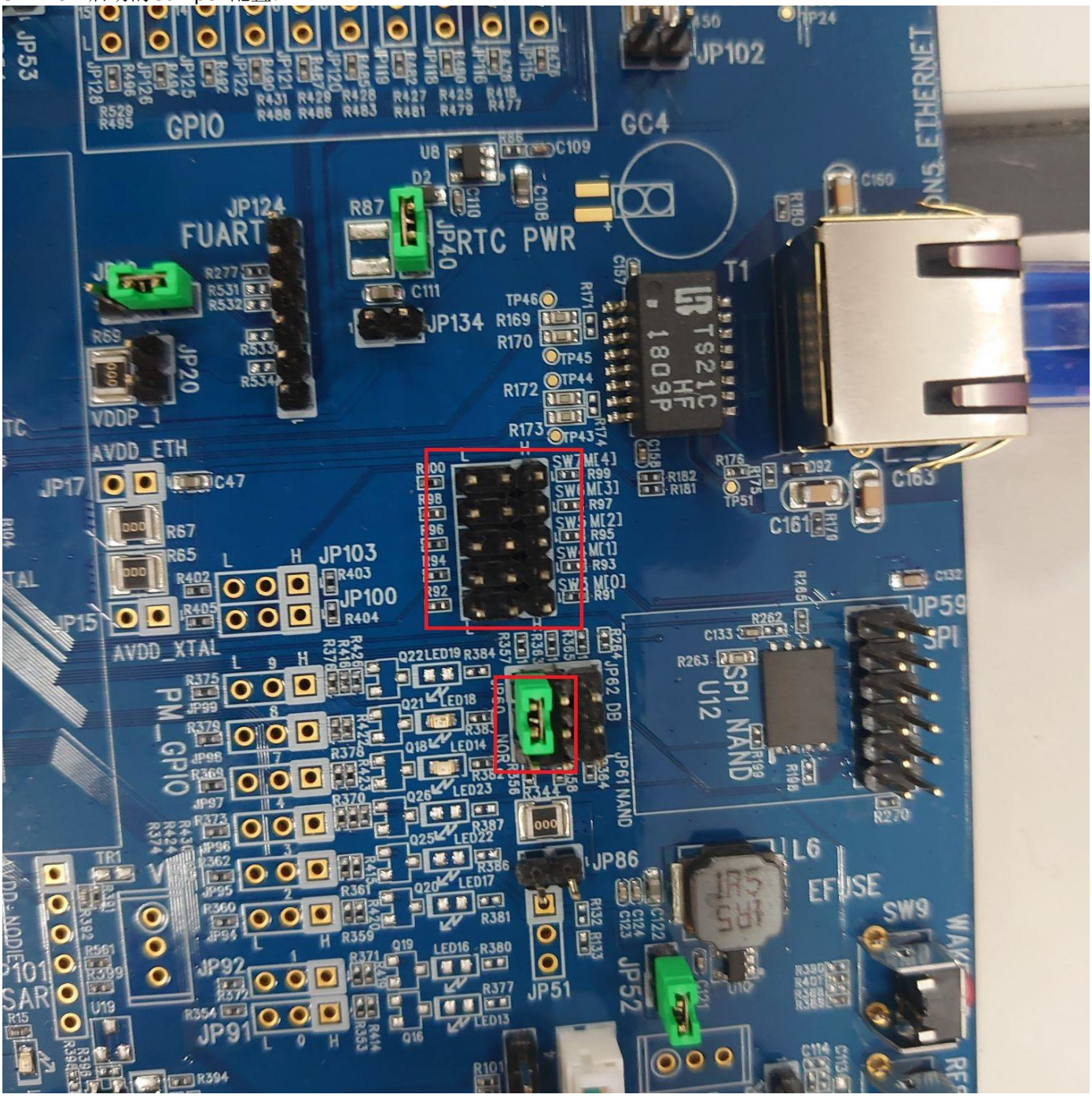

Figure 4 图 1-5

SPI-NAND 启动的 Jumper 配置

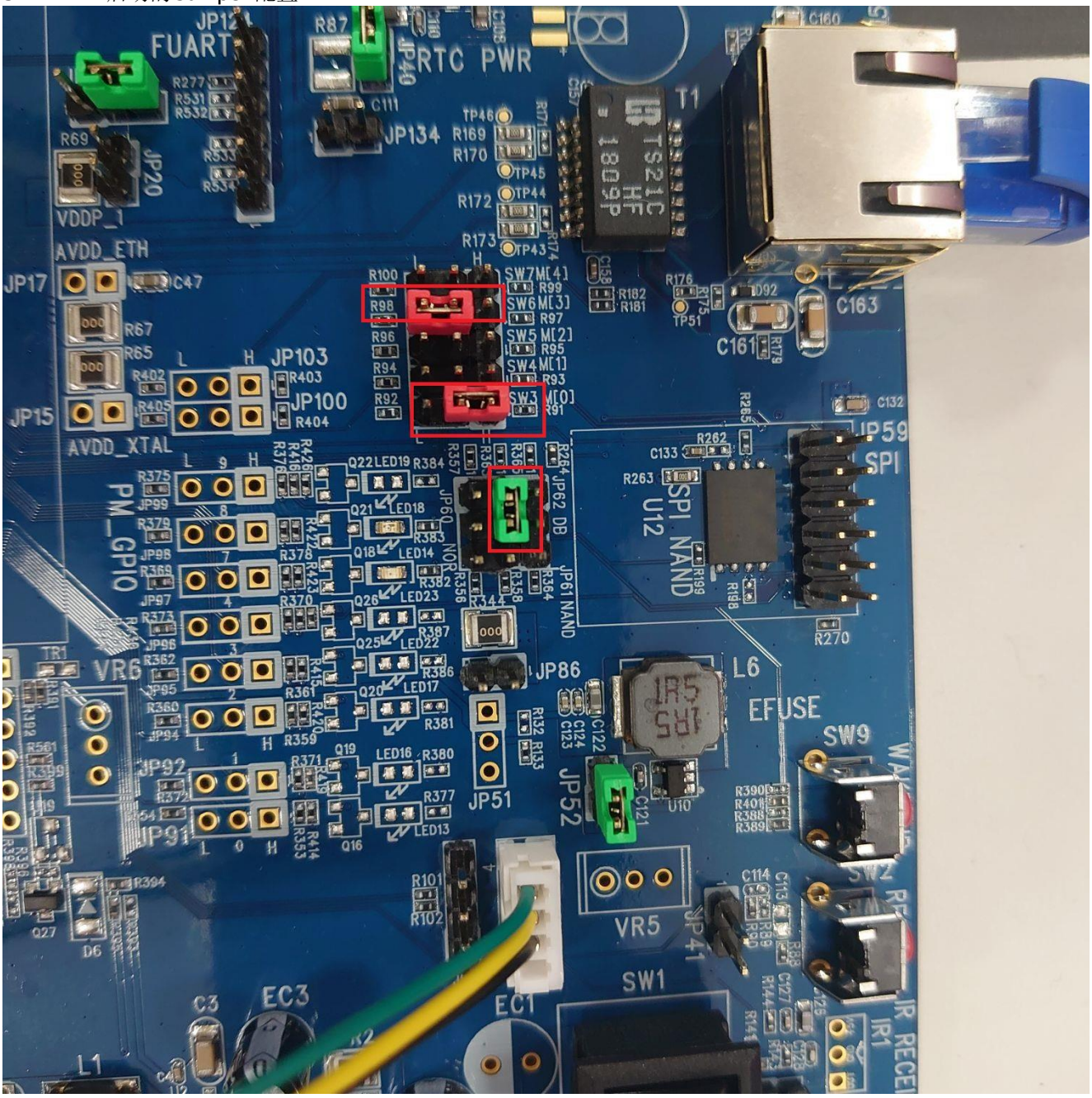

Figure 5 图 1-6

2. 准备编译环境

通常我们会以交叉编译的方式进行开发和调试，即“宿主机+目标机”的形式。而宿主机和目标机的连接我们一般采用串口连接或网络连接，如下图所示：

Figure 6: 图 2-1 交叉编译连接方式

注：我们提供的 debug tool 作用为读寄存器和烧写 Mboot。

2.1. 安装 Linux 服务器

建议使用 Ubuntu 16.04。

2.2. 安装交叉编译工具

我们提供编译工具链 gcc-linaro-arm-linux-gnueabihf-4.8-2014.04_linux 用于编译 glibc 版本的 mi 和 kernel, boot，arm-buildroot-linux-uclibcgnueabihf-4.9.4 用于 uclibc 版本，工具链一般打包在 SDK 开发包中，或者请找 FAE 索取。

3. 编译

本芯片支持 nor flash 和 spi nand flash 两种启动方式，因此在 SDK 中编译有所区分，通过不同的配置文件来实现，SDK 中包含了最新的 boot 和 kernel 的 image，并且脚本打包成可烧录的 image。

3.1.1 编译 boot

- **SPI-NOR package**

```
#declare -x ARCH="arm"
#declare -x CROSS_COMPILE="arm-linux-gnueabihf-"
# make infinity6b0_defconfig;
#make clean;
#make
```

- **SPI-NAND package**

```
#declare -x ARCH="arm"
#declare -x CROSS_COMPILE="arm-linux-gnueabihf-"
# make infinity6b0_spinand_defconfig;
#make clean;
#make
```

- **Get image**

```
# cp u-boot.xz.img.bin ${ your_release_path }  // 选择 spi-nor 时
# cp u-boot_spinand.xz.img.bin ${ your_release_path }    // 选择 spi-nand 时
```

3.1.2 编译 kernel

- **SPI-NOR Kernel (ASIC)**

CHIP	Glibc compiler	Uclibc compiler	Kernel make config	DTS
QFN88 64MB: SSC335 QFN88 128MB: SSC335D	Linaro Glibc8.2.1 arm-linux-gnueabihf-	Buildroot Uclibc 4.9.4 arm-buildroot-linux-uclibcgnueabihf-	infinity6b0_ssc009a_s01a_defconfig	infinity6b0-ssc009a-s01a.dts
QFN128 128MB: SSC337DE	Linaro Glibc8.2.1 arm-linux-gnueabihf-	Buildroot Uclibc 4.9.4 arm-buildroot-linux-uclibcgnueabihf-	infinity6b0_ssc009b_s01a_defconfig	infinity6b0-ssc009b-s01a.dts

Table 1

- **SPI-NAND Kernel (ASIC)**

CHIP	Glibc compiler	Uclibc compiler	Kernel make config	DTS
QFN88 64MB: SSC335 QFN88 128MB: SSC335D	Linaro Glibc8.2.1 arm-linux-gnueabihf-	Buildroot Uclibc 4.9.4 arm-buildroot-linux-uclibcgnueabihf-	infinity6b0_ssc009a_s01a_spinand_defconfig	infinity6b0-ssc009a-s01a.dts
QFN128 128MB: SSC335DE	Linaro Glibc8.2.1 arm-linux-gnueabihf-	Buildroot Uclibc 4.9.4 arm-buildroot-linux-uclibcgnueabihf-	infinity6b0_ssc009b_s01a_spinand_defconfig	infinity6b0-ssc009b-s01a.dts

Table 2

注：请参考表格和你的芯片版本做对应的编译。

```
# declare -x ARCH="arm"
# declare -x CROSS_COMPILE="$compiler"
      //exp: uclibc "arm-buildroot-linux-uclibcgnueabihf"
# make xxx_kernel_make_config
      //exp: make infinity6b0_ssc009b_s01a_spinand_defconfig
# make clean;
# make
```

- **Get image**

```
# cp arch/arm/boot/uImage.xz ${ your_release_path }
```

3.1.3 编译 SDK(ALKAID)

- **SPI-NOR flash package**

CHIP	Glibc	Uclibc
QFN88 64MB: SSC335	nor.glibc-squashfs.009a.64.qfn88	nor.uclibc-squashfs.009a.64.qfn88
QFN88 128MB: SSC335D	nor.glibc-squashfs.009a.128.qfn88	nor.uclibc-squashfs.009a.128.qfn88
QFN128 128MB: SSC337DE	nor.glibc-squashfs.009b.128.qfn128	nor.uclibc-squashfs.009b.128.qfn128

3Table 3

- **SPI-NAND flash package**

CHIP	Glibc	Uclibc
QFN88 64MB: SSC335	spinand.glibc-squashfs.009a.64.qfn88	spinand.uclibc-squashfs.009a.64.qfn88
QFN88 128MB: SSC335D	spinand.glibc-squashfs.009a.128.qfn88	spinand.uclibc-squashfs.009a.128.qfn88
QFN128 128MB: SSC337DE	spinand.glibc-squashfs.009b.128.qfn128	spinand.uclibc-squashfs.009b.128.qfn128

Table 4

注：请参考表格和你的芯片版本做对应的编译。

```
# cd ${Alkaid}/project
# . /setup_config.sh xxx_alkaid_build_config
//exp: . /setup_config.sh ./configs/ipc/i6/nor.glibc-squashfs.009a.128.qfn88
# make image
```

- **Get image**

```
# cd ${Alkaid}/project/image/output/images
```

4. 烧写

4.1.1 Burning code by uboot

- **Run tftp (FTP server) on PC**

Step1. 使用 tftp 工具指向 image path：SDK\project\image\output\images\，并选择正确的网卡。

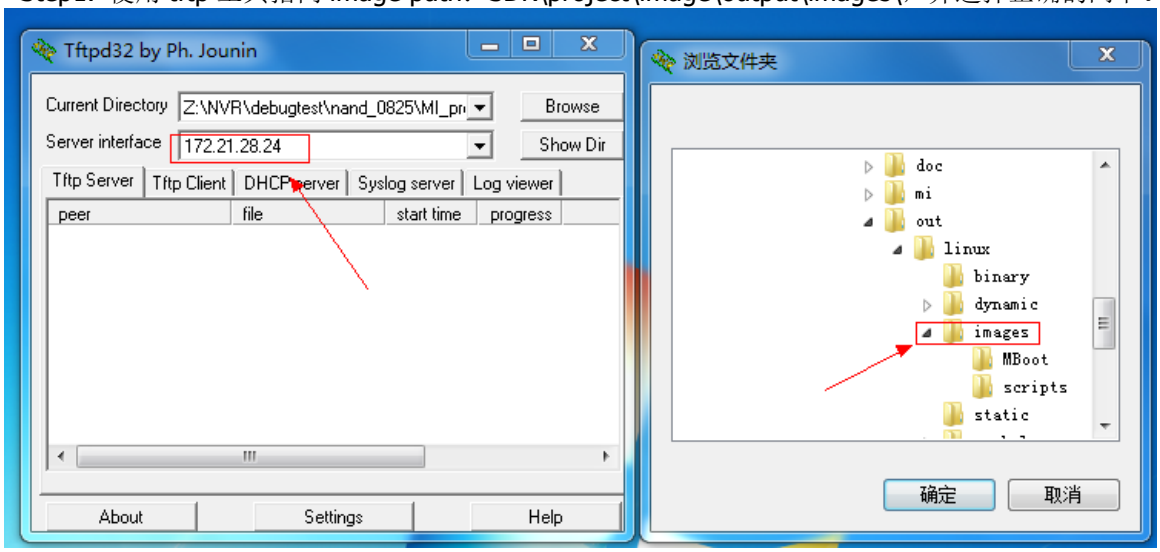

Figure 7: 图 4-1

Step 2. 连接板子的网口到 PC 端，连接 debug 串口工具到 PC 端，并检查连接的正确性。

- **Run tftp (FTP Client) on EVB**

Step 1. 板子开机，长按回车，进入 bootloader command line. Mstar 的 boot loader。

- 首次烧录请设置 IP：(除非 ip 设置变更或者更换 flash)

```
# set -f gatewayip 192.168.1.1
# set -f ipaddr 192.168.1.127              //设定 FTP Client (EVB 板子)使用的 IP
# set -f netmask 255.255.255.0
# set -f serverip 192.168.1.100          //设定 FTP server (PC) 的 IP
# saveenv
```

注：

1. 为了保证烧录顺利，请保证 PC 和开发板处于同一网段。
2. 请采用静态方式固定分配 ip。防止烧录时 ip 地址跳变。
3. 您也可以使用独立网卡使 PC 端直连开发板，固定该网卡的内网 ip 地址，并按上述方法设定开发板。

- 在 UBOOT console 下执行以下指令即可自动透过 ethernet 烧录。

```
# estar (OR: estar auto_update.txt)
```

4.1.2 Burning uboot by ISP Tool

本方式适用于空机烧录，或者 uboot 已经损坏导致无法通过 uboot 升级的场合。

4.1.2.1. SPI-NOR-Flash

- **Default Partition layout**

No	range	size
IPL	0x00000000, 0x00010000	64KB
IPL_CUST	0x00010000, 0x0000F000	60KB
KEY_CUST	0x0001F000, 0x00001000	64KB
MXPT	0x00020000, 0x00001000	64KB
UBOOT	0x00030000, 0x0001F000	124KB
UBOOT_ENV	0x0004F000, 0x00001000	64KB
BOOT	0x00000000, 0x00050000	320KB
KERNEL	0x00050000, 0x00200000	2048KB
ROOTFS	0x00250000, 0x00400000	4096KB
NVRSERVICE	0x00650000, 0x00300000	3072KB
CUSTOMER	0x00950000, 0x006B0000	6848KB

Table 5

- **Burning code by ISP tool**

	offset	Binary 放置目录
IPL.bin	0x0000	${ALKAID}\project\image\output\images\IPL.bin
IPL_CUST.bin	0x10000	${ALKAID}\project\image\output\images\IPL_CUST.bin
MXP_SF.bin	0x20000	${ALKAID}\project\image\output\images\MXP_SF.bin
u-boot.xz.img.bin	0x30000	${ALKAID}\project\image\output\images\u-boot.xz.img.bin

Table 6

- **Burning Steps**
  - Step1. 执行 ISP tool。并且关闭 UART terminal，否则无法正常'Connect'。
  - Step2. 选择 SPI tab，点击'More' 并且选择类型为'SPI'。

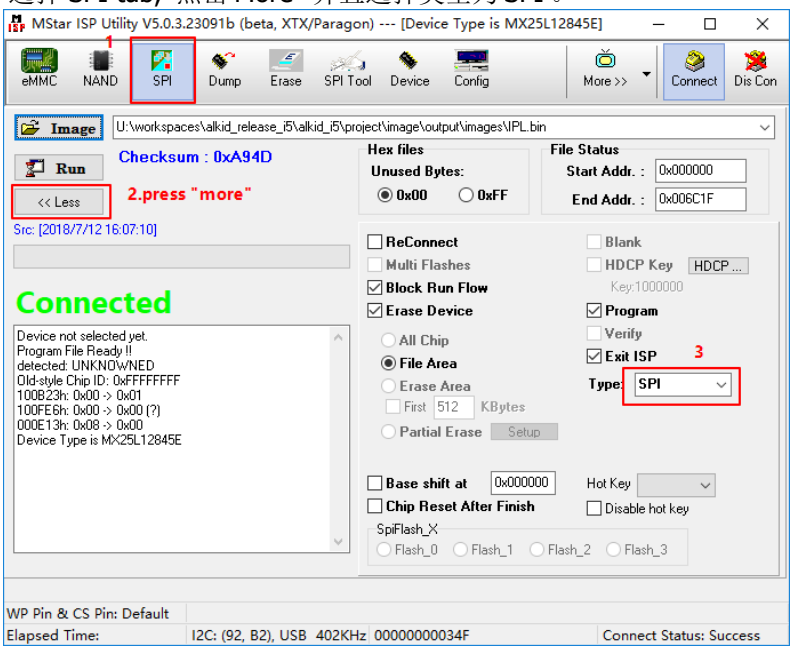

Figure 8: 图 4-2

- Step3. 加载烧录文件并点击'Connect'。

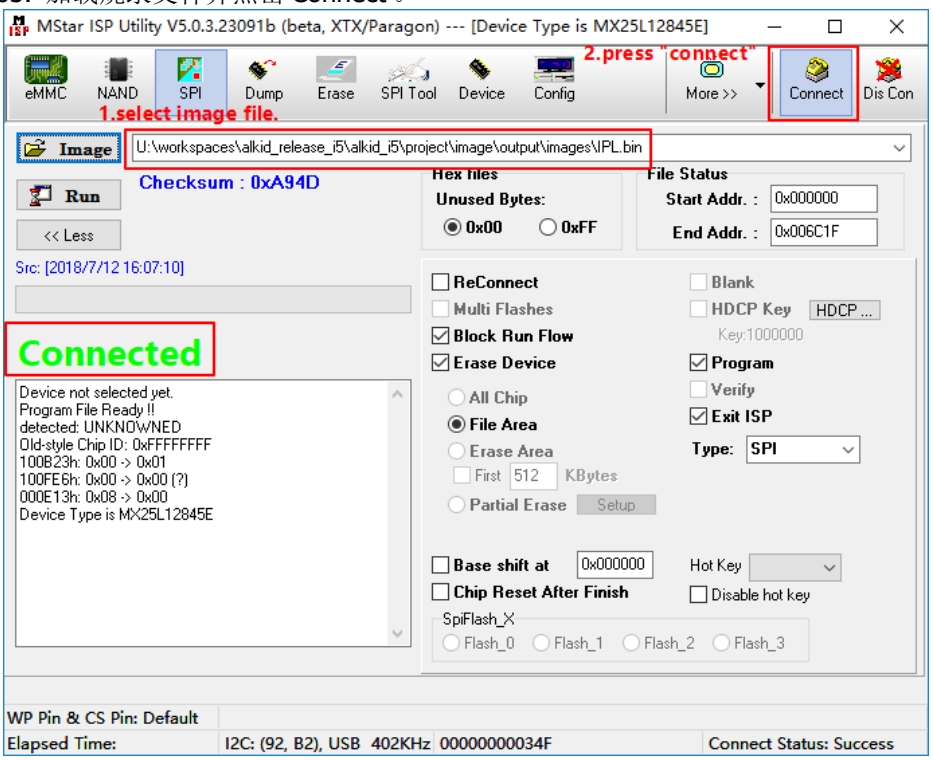

Figure 9: 图 4-3

- Step4. 加载 image "IPL.bin",并点击'Run'。

注：需要勾选' erase file area'。

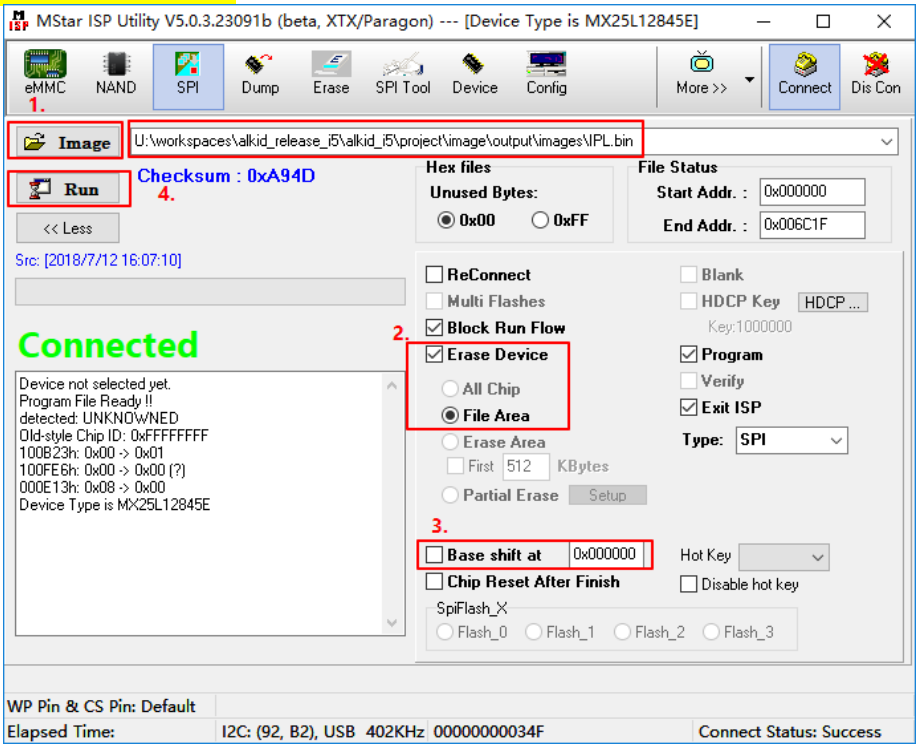

Figure 10 图 4-4

- Step5. 加载 image "IPL_CUST.bin", 取消'Erase Device'选项。设置'Base shift 'at 0x10000。

注:可能随着版本变化，Base shift 的地址以表格 1-2 数据为准。

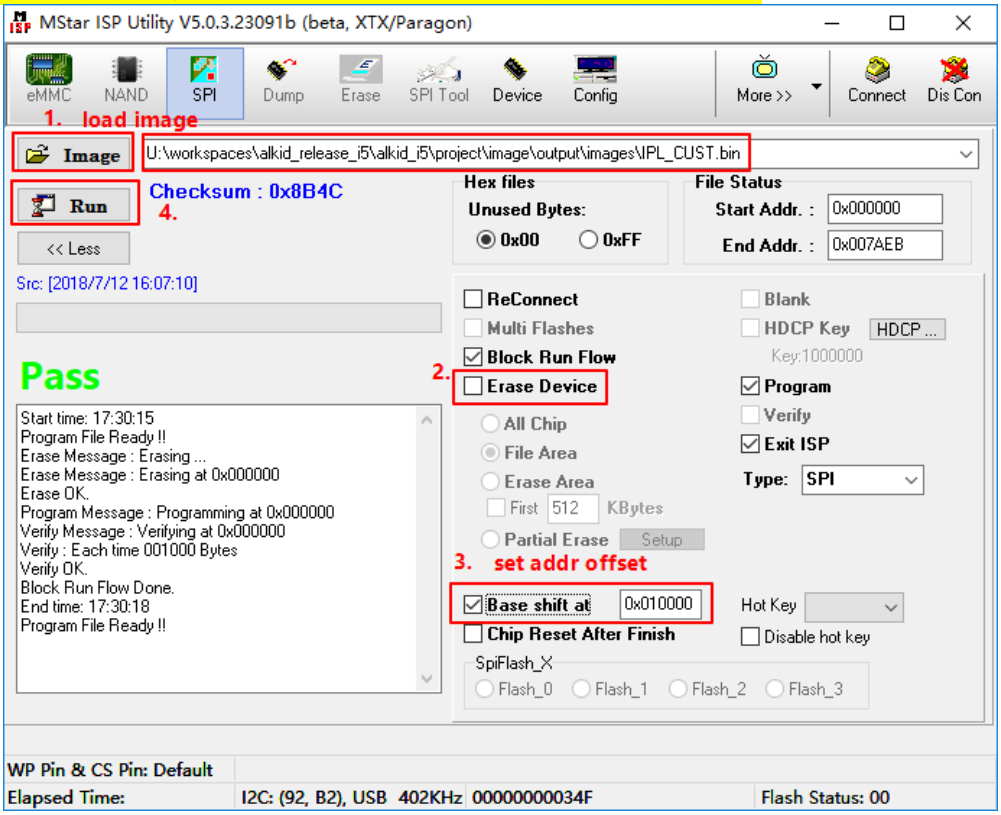

Figure 11: 图 4-5

- Step6. 加载 image "MXP_SF.bin", 设置'Base shift 'at 0x20000。

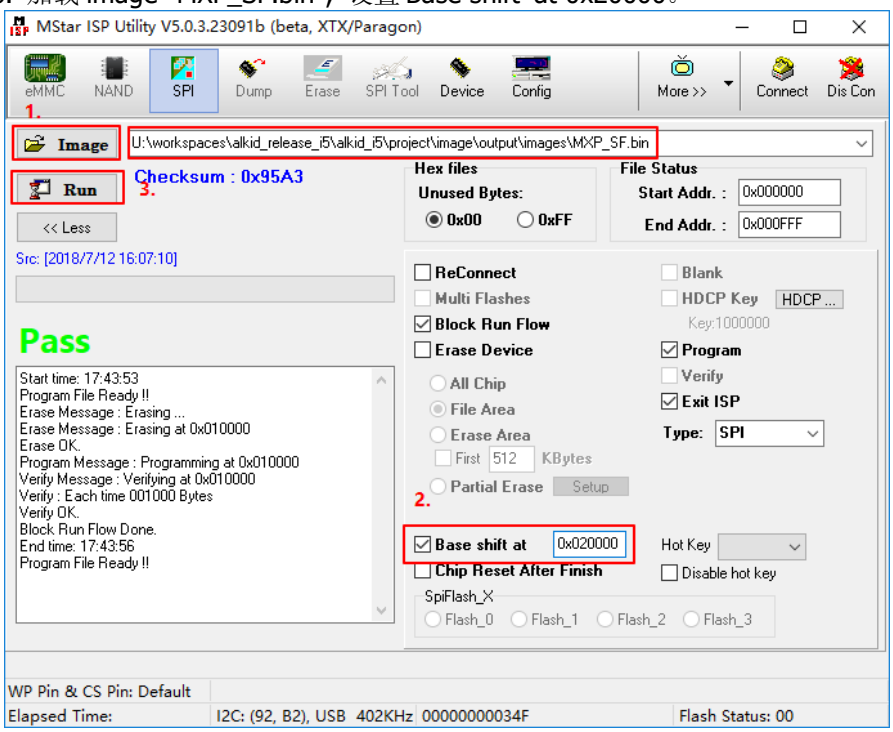

Figure 12: 图 4-6

- Step7. 加载 image “u-boot.xz.img.bin”，设置‘Base shift ‘at 0x30000。

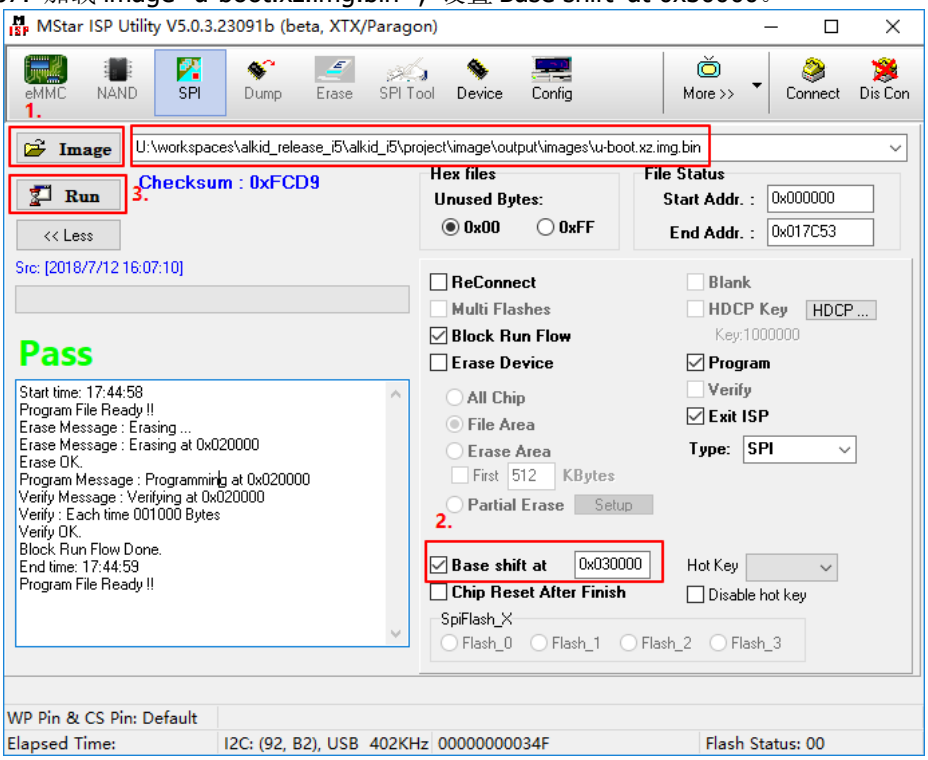

Figure 13: 图 4-7

- Step8. 重启 EVB 板子，关闭工具。

4.1.2.2. SPI-NAND Flash

- **Default Partition layout**

No	range	size
CIS	0x00000000-0x0060000	384KB
IPL0	0x00060000-0x0080000	128KB
IPL1	0x00080000-0x00A0000	128KB
IPL2	0x000A0000-0x00C0000	128KB
IPL_CUST0	0x000C0000-0x00E0000	128KB
IPL_CUST1	0x000E0000-0x0100000	128KB
IPL_CUST2	0x00100000-0x0120000	128KB
UBOOT0	0x00120000-0x0180000	384KB
UBOOT1	0x00180000-0x01E0000	384KB
ENV	0x001E0000-0x0200000	384KB
KERNEL	0x00200000-0x0700000	5120KB
RECOVERY	0x00700000-0x0C00000	5120KB
UBI	0x00C00000-0x008000000	112640KB

Table 7

- **Burning code by ISP tool**
  - **ISP Tool Version**

    请确定 ISP Tool 版本为 V5.0.3.23091b(beta)。

    ISP Tool 会在首次版本发布的时候一起打包在 Tool 目录下。

  - **Images list**

	Offset	**Image 所在目录**
GCIS.bin	0x000000	project\image\output\images\ GCIS.bin
IPL.bin	0x60000	project\image\output\images\IPL.bin
IPL_CUST.bin	0xC0000	project\image\output\images\IPL_CUST.bin
u-boot_spinand.xz.img.bin	0x120000	project\image\output\images\ u-boot_spinand.xz.img.bin

Table 8

  - **Burning Steps**
    - Step1. 执行 ISP tool。并且关闭 UART terminal，否则可能无法正常'Connect'。

Figure 6: 图 1-6

    - Step2. 选择 SPI tab，点击'More' 并且选择类型为'SPINAND'。

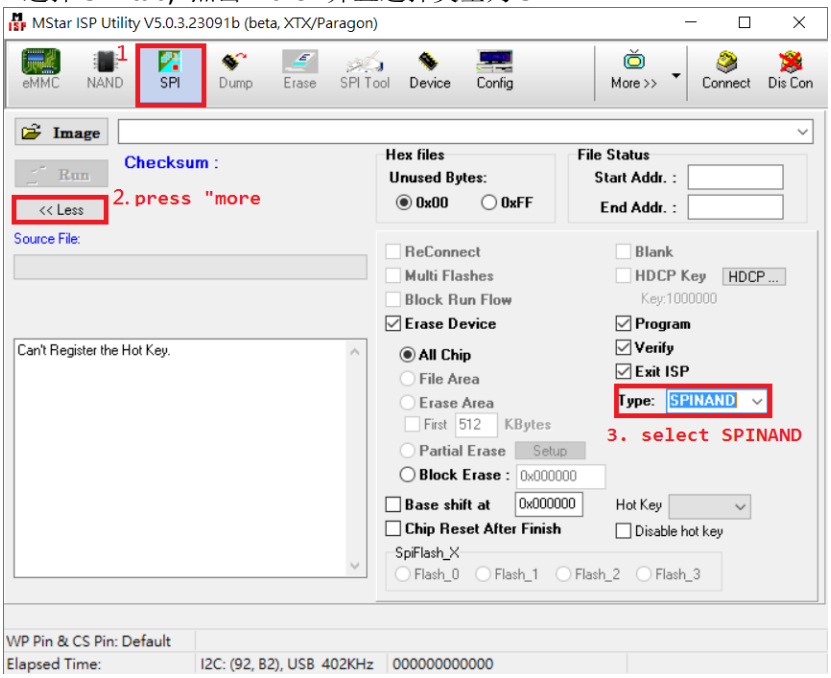

Figure 14: 图 4-8

    - Step3. 加载烧录文件并点击'Connect'。

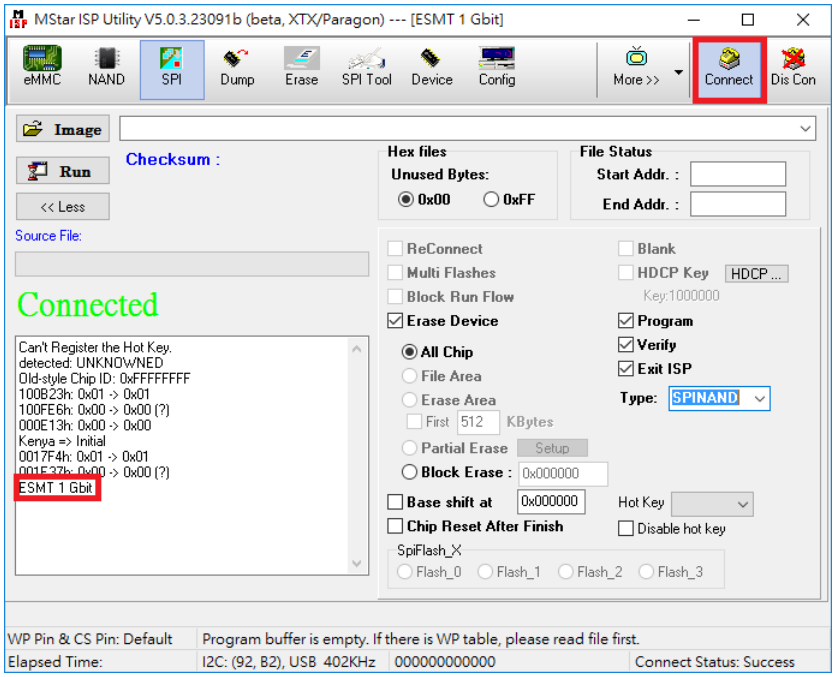

Figure 15: 图 4-9

- Step4. 加载 image "GCIS.bin",并点击'Run'
  注：需要勾选' erase all chip'

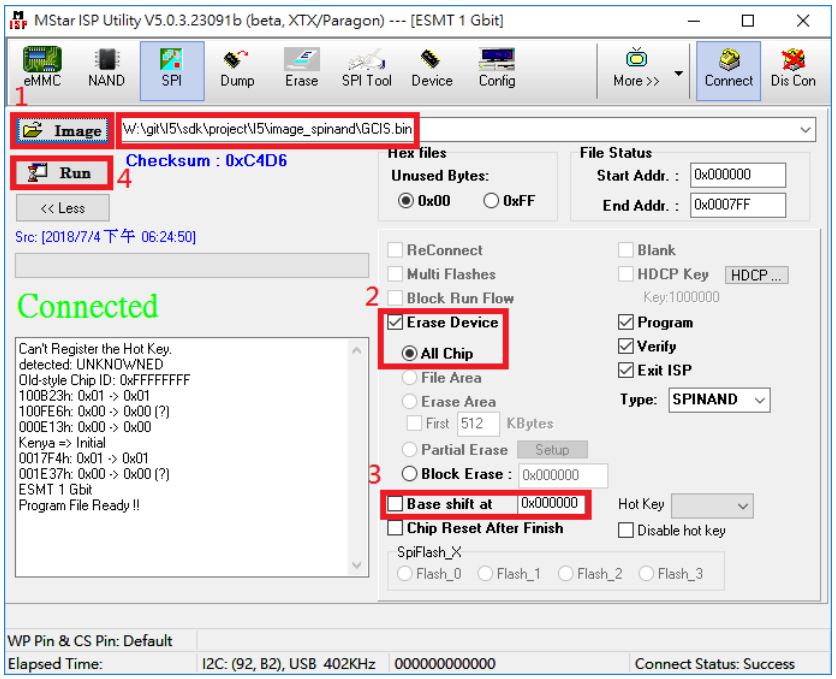

Figure 16: 图 4-10

- Step5. 加载 image "IPL.bin", 取消'Erase Device'选项, 设置'Base shift 'at 0x140000。

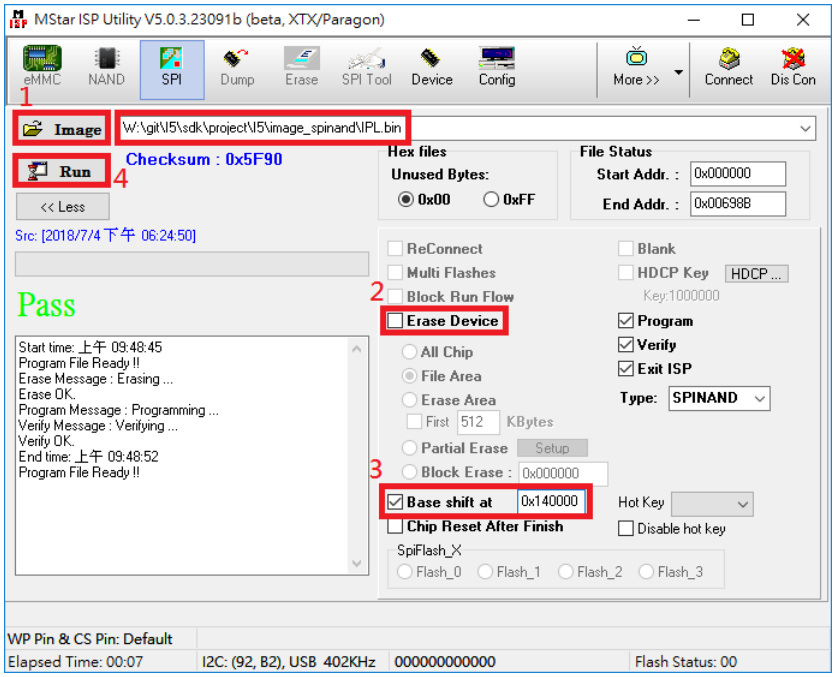

Figure 17: 图 4-11

- Step6. 加载 image “IPL_CUST.bin”，设置‘Base shift ‘at 0x200000。

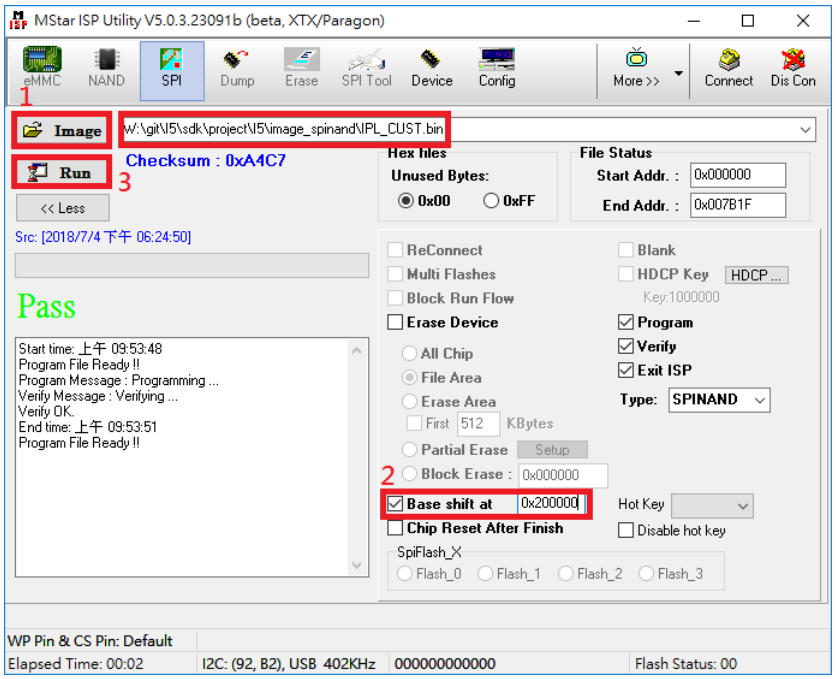

Figure 18: 图 4-12

- Step7. 加载 image “u-boot_spinand.xz.img.bin”，设置‘Base shift ‘at 0x2C0000。

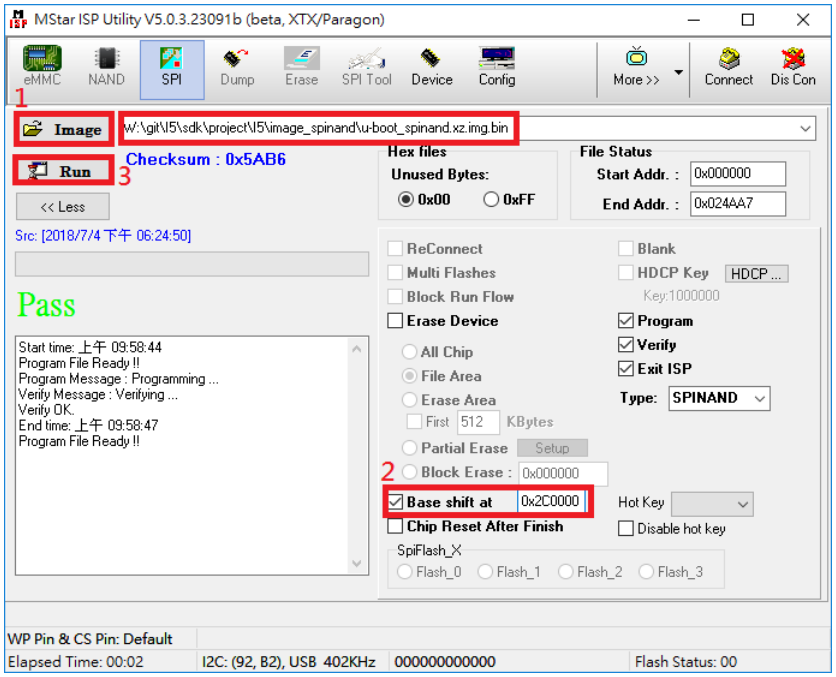

Figure 19: 图 4-13

- Step8. 重启 EVB 板子，关闭工具。