Hikey970

Development Guide

Issue **01**

Date **2018-03-15**

HiSilicon Technologies Co., Ltd.

Address: Huawei Industrial Base
Bantian, Longgang
Shenzhen 518129
People's Republic of China

Website: http://www.hisilicon.com/cn/

Email: support@hisilicon.com

About This Document

Purpose

This document instructs developers to compile and upgrade the Hikey970 native kernel image.

Intended Audience

This document is intended for:

- Instruct developers to compile native kernel images;
- Guide the developer to upgrade kernel images.

Change History

Changes between document issues are cumulative. The latest document issue contains all the changes made in earlier issues.

Issue 01 (2018-03-15)

This issue is used for first office application (FOA).

Contents

1 Compiling Userspace

1.1 Download the Android Source Tree

```
repo init -u https://android.googlesource.com/platform/manifest -b master
git clone https://github.com/96boards-hikey/android-manifest.git -b
hikey970_v1.0 .repo/local_manifests
repo sync --force-sync -j8
```

1.2 Repo Sync

Copy kirin970-hikey970.dtb.dtb (arch/arm64/boot/dts/hisilicon/ kirin970-hikey970.dtb) to the device/linaro/hikey-kernel directory as file: kirin970-hikey970.dtb-4.9

Copy the Image file (arch/arm64/boot/Image.gz-dtb) to the device/linaro/hikey-kernel directory as file: Image.gz-hikey970-4.9

1.3 Build System Image

```
source ./build/envsetup.sh
lunch hikey970-userdebug
make systemimage -j32
```

2 Compiling Boot Image

2.1 Download the Code of Hikey970 on GitHub

Git command:

```
git clone https://github.com/96boards-hikey/linux.git
```

Note: after the code is downloaded, switch to the hikey970 branch (Branch Name: origin/hikey970-v4.9).

2.2 Prepare Tools and Files

2.2.1 Create a Tools Folder

Create a “tools” folder in the same level directory of the “linux” directory.

```
./linux/arch
./linux/mm
./linux/kernel
…
./tools/
```

2.2.2 Copy ramdisk.img and mkbootimg to the Tools Directory

Copy the compiled ramdisk.img and mkbootimg into the tools directory.

“mkbootimg” is under the “system/core/mkbootimg” directory of the AOSP code.

After compiling userspace (Please refer to chapter 1), “ramdisk.img” is under the “/out/target/product/hikey970” directory.

2.2.3 Modify the Compilation Script

Create the “build_kernel.sh” file, then copy the contents of the following text box and save it.

```
#!/bin/bash
DTB=1
LOCAL_DIR=$(pwd)
KERNEL_DIR=${LOCAL_DIR}/linux
PRODUCT_OUT=${LOCAL_DIR}/out/target/product/hikey970
GEN_IMAGES_DIR=${LOCAL_DIR}/tools
HIKEY970_KERNEL=${LOCAL_DIR}/tools
```

```
MKBOOTTOOL_DIR=${LOCAL_DIR}/tools
CURRENT_DIR=${LOCAL_DIR}
NCPU=`grep -c ^processor /proc/cpuinfo`

if [ ! -e ${PRODUCT_OUT} ]
then
    mkdir -p ${PRODUCT_OUT}
fi

export MINI_SYS=true
export ARCH=arm64
export
CROSS_COMPILE=~/myview/code/hikey970_trunk_new/prebuilts/gcc/linux-x86/aa
rch64/aarch64-linux-android-4.9/bin/aarch64-linux-android-

function check_build_result()
{
    if [ $? != 0 ]; then
        echo -e "\033[31m $1 build fail! \033[0m"
        exit -1
    else
        echo -e "\033[32m $1 build success! \033[0m"
    fi
}

if [ "${MINI_SYS}" != true ]; then
    source  ./build/envsetup.sh && lunch hikey970-userdebug && make
-j$[NCPU*2] $2
    check_build_result "Android System"
fi

cd ${KERNEL_DIR}

make hikey970_defconfig && \
make -j$[NCPU*2] Image.gz modules

check_build_result "Kernel Image"
rm -f arch/arm64/configs/hikey970_temp_defconfig

cp arch/arm64/boot/Image.gz ${HIKEY970_KERNEL}
if [ $DTB -eq 1 ]; then
    make hisilicon/kirin970-hikey970.dtb
    check_build_result "Hikey970 dtb"
    cp arch/arm64/boot/dts/hisilicon/kirin970-hikey970.dtb
```

```
${HIKEY970_KERNEL}
fi

cd ${CURRENT_DIR}

if [ ${need_repack_userdata} ];
then
    make -j$[NCPU*2] userdataimage-nodeps
    check_build_result "Hikey970 need repack userdataimage"
fi

if [ "${MINI_SYS}" = true ]; then
    RAMDISK=${GEN_IMAGES_DIR}/ramdisk.img
else
    RAMDISK=${PRODUCT_OUT}/ramdisk.img
    if [ ! -e $RAMDISK ]; then
        echo -e "\033[33m $RAMDISK is not exist! please build ramdisk first.
\033[0m"
        echo -e "\033[33m . ./build/envsetup.sh && lunch hikey960-userdebug
&& make ramdisk \033[0m"
        exit -1
    fi
fi

#uefi boot.img = Image + dtb + ramdisk
cat ${KERNEL_DIR}/arch/arm64/boot/Image
${KERNEL_DIR}/arch/arm64/boot/dts/hisilicon/kirin970-hikey970.dtb >
${HIKEY970_KERNEL}/Image-dtb
check_build_result "Image-dtb"

${MKBOOTTOOL_DIR}/mkbootimg --kernel ${HIKEY970_KERNEL}/Image-dtb --ramdisk
${RAMDISK} --cmdline "androidboot.hardware=hikey970
firmware_class.path=/system/etc/firmware loglevel=15
buildvariant=userdebug androidboot.selinux=permissive
clk_ignore_unused=true" --base 0x0 --tags_offset 0x07A00000 --kernel_offset
0x00080000 --ramdisk_offset 0x07c00000 --os_version 7.0 --os_patch_level
2016-08-05  --output ${PRODUCT_OUT}/boot.img
check_build_result "Boot Image"

echo -e "\033[36m build boot.img complete! \033[0m"
```

Edit the compile script CROSS_COMPILE parameters to specify the compiler tool according to your own compilation environment.

```
export
```

```
CROSS_COMPILE=/home/xxxxxx/hikey970/prebuilts/gcc/linux-x86/aarch64/aarch
64-linux-android-4.9/bin/aarch64-linux-android-
```

2.3 Compile

2.3.1 Run build_kernel.sh

Execute "./build_kernel.sh" in the tools directory to compile the image. After the compilation is completed, the boot.img can be generated under “out/target/product/hikey970”.

3 Base Firmware Files and Installation

This section shows how to install all base firmware components for the HiKey970. Once finished with these instructions, please continue to the HiKey970 documentation landing page to flash an operating system.

- **Step 1**: Material and preparation
- **Step 2**: Dependencies
- **Step 3**: Enter recovery/forced-download mode on HiKey970
- **Step 4**: Flash base firmware
- **Step 5**: Explore other modes, proceed to OS installation
- **Troubleshooting**

3.1 Step 1: Material and Preparation

- HiKey970
- USB Type-A (Host machine) to USB Type-C (96Boards) cable
- 96Boards compliant power supply
- To boot into fastboot mode everytime set switch 1 & 3 to ON state and switch 2 to OFF state.
- To boot into fastboot mode at every alternate reboot set switch 1 to ON and switch 2 & 3 to OFF state.
- To boot into recovery mode set switch 1 & 2 & 3 to ON state

3.2 Step 2: Dependencies

Host Linux Machine

- Remove modem manager. At least in Ubuntu 14.04 and 16.04 version, we found a conflicting issue if modem manager is installed and active. Modem manager monitors ttyUSBx's incoming data, when it reads some given pattern, it will send some bytes back into the tty as response. And those bytes sent by modem manager can make board side recovery flashing tool confuse and fail. Solution is to uninstall this service. If you have a

doubt whether you are safe to remove it or not, double confirm here: ModemManager homepage.

```
$ sudo dpkg -s modemmanager
$ sudo apt-get remove modemmanager
```

- Android SDK “Platform-Tools” for Linux can be downloaded here
- Use terminal to clone this repository into desired folder and cd into tools-images-HiKey970

```
$ git clone https://github.com/96boards-hikey/tools-images-hikey970.git
$ cd tools-images-hiKey970
```

3.3 Step 3: Enter recovery/forced-download Mode on HiKey970

- Remove power from the board
- Change Jumper/DIP switch settings, to enter recovery/forced-download mode:

Name	Switch	State
Auto Power up	Switch 1	ON
Recovery	Switch 2	ON
Fastboot	Switch 3	ON

- Apply power to the board using 96Boards compliant power supply
- Insert USB Type-C cable (OTG port) to the board, and connect the other end to your Linux PC
- Check whether there is a device node "/dev/ttyUSBx". If there is, it means your PC has detected the target board; If there is not, try to repeat previous steps.

3.4 Step 4: Flash Base Firmware

Once again using the terminal on your host machine, execute the following command. Be sure to replace /dev/ttyUSBx with the USB value detected by your machine.

```
$ sudo ./recovery-flash.sh /dev/ttyUSBx
```

After it completes, the base firmware will be flashed to the device, this does not mean OS.

The board will then be in fastboot mode.

3.5 Step 5: Explore Other Modes, Proceed to OS Installation

- sw2402 mode
- Proceed to OS "Installation" through the HiKey970 documentation landing page

3.6 Troubleshooting

- If recovery script "./recovery-flash.sh /dev/ttyUSBx" fail to run to completion and you see "< waiting for any device >" in a loop, then try uninstalling modem manager from your host machine. The script will work after that. Don't forget to install modem manager back after recovery.

Switch	Normal Mode	Fastboot Mode	Recovery Mode
Switch 1	ON	ON	ON
Switch 2	OFF	OFF	ON
Switch 3	OFF	ON	ON

- If you run into trouble, see the README-technical.mdfile in this directory.

4 Hisi-idt

A tool for downloading binaries to soc RAM and DDR through serial port.

4.1 Command

- Linux

```
sudo python hisi-idt.py -d /dev/ttyUSBx --img1 ./ sec_usb_xloader.img
--img2 ./sec_usb_xloader2.img --img3 ./l-loader.bin
```

- Windows

```
python hisi-idt.py -d commxx --img1 sec_usb_xloader.img
--img2 sec_usb_xloader2.img --img3 l-loader.bin
```

4.2 Download Steps

Step 1 Insert USB cable and connect with PC;

Step 2 Enter force download mode:

For hikey970 board: sw2402

switch 1 mode: ON

switch 2 mode: ON

switch 3 mode: ON

release “Reset” key then will enter into “force download” mode;

Step 3 Check if there have the device node “/dev/ttyUSBx”, if there have device node that means the PC has detected the target board; d. Use command "sudo python hisi-idt.py" to run the script; after IDT download binaries successfully, it will print out below log:

```
+----------------------+
 Serial:  /dev/ttyUSB1
 Image1:  fastboot1.img
 Image2:  fastboot2.img
```

```
+----------------------+

Sending fastboot1.img ...
Done

Sending fastboot2.img ...
Done
```

4.3 Burn Images

After download fastboot1.img and fastboot2.img on the board, then can use fastboot command to burn images:

```
sudo fastboot flash fip  fip.bin
sudo fastboot flash fastboot l-loader.bin
```

4.4 Troubleshooting

Step 1 After enter the force download mode, if Ubuntu PC cannot recognize the device ttyUSBx; this issue can be fixed by input below commands: sudo echo 12D1 3609 > /sys/bus/usb-serial/drivers/option1/new_id sudo makenod /dev/ttyUSB0 c 188 0

Step 2 Need supervisor permission for hisi-idt.py: "sudo python hisi-idt.py"

Step 3 Need supervisor permission for fastboot: "sudo fastboot"

Step 4 If download binaries failed with below message:

```
Sending fastboot1.img ...
failed
failed
```

Usually this means you are using the wrong ttyUSBx device; the reason is when connect board with the UART cable and USB cable, then PC will create two device nodes /dev/ttyUSB0 and /dev/ttyUSB1;But the nodes which are randomly binding to UART and USB, so sometimes /dev/ttyUSB0 is created for the UART and /dev/ttyUSB1 is for the USB port, in this case should use /dev/ttyUSB1 for the IDT; if PC exchanges the nodes then should use /dev/ttyUSB0.