AN14113

如何在i.MX 8M Nano和i.MX 93上实现U-Boot Falcon模式

第2版—2024年2月22日 应用笔记

文档信息

信息	内容
关键词	AN14113、i.MX 8M、i.MX 93、U-Boot、Falcon、快速启动
摘要	本应用笔记介绍了实现Falcon模式的方法。

1 介绍

i.MX 8M系列应用处理器基于Arm Cortex-A53和Cortex-M内核，为各种应用提供业界领先的音频、语音和视频处理能力，适用范围从消费级家庭音响到工业楼宇自动化，乃至移动计算机。

i.MX 93应用处理器提供高效的机器学习（ML）加速和先进的安全性，其集成的EdgeLock安全区域支持高能效的边缘计算。

Falcon模式是U-Boot的一项功能，它允许系统直接从二级程序加载器（SPL）启动Linux内核，加速启动过程。

默认的i.MX版本支持使用启动命令（`booti`、`bootz`或`bootm`等）从U-Boot启动内核。该过程的一个缺点是内核加载相对较晚。

使用Falcon模式，我们可以缩短这一启动时间。当用户需要内核的快速启动功能时，Falcon模式尤为有用。

本应用笔记详细介绍了Falcon模式的实现方法。

本文档的目标受众包括以下人员：

1. 需要在系统早期启动内核的用户。
2. 希望深入了解ROM API使用方法的人员。
3. 想要掌握imx-mkimage使用技巧的人员。
4. 对SPL启动机制感兴趣的人员。

本应用笔记中所述的机制也同样适用于其他i.MX 8M和i.MX 9系列芯片。

2 定义、缩略语与缩写

表1. 定义、缩略语和缩写

缩略语	含义
ATF	Arm Trusted Firmware Arm可信固件
DTB	Device Tree Blob设备树Blob
DTS	Device Tree Source设备树源码
FIT	Flattened Image Tree扁平映像树
OS	Operating System操作系统
ROM	Read-Only Memory只读存储器
SoC	System on Chip片上系统
SPL	Secondary Program Loader二级程序加载器

3 U-Boot SPL简介

SPL是U-Boot的精简版。它有助于进行一些初始硬件配置，例如使用CPU缓存作为RAM进行DRAM初始化，并加载更大、功能更全面的U-Boot版本。

电路板启动时，除了ROM代码外，*第一阶段引导加载程序*是U-Boot SPL；而*第二阶段引导加载程序*是常规U-Boot（或U-Boot本身）。需要明确的是：SPL是二级程序加载器（Secondary Program Loader）的缩写，意思是ROM代码是加载和执行另一个程序的第一件事，而SPL是加载和执行另一个程序的第二件事。

图1显示了常见的启动顺序。

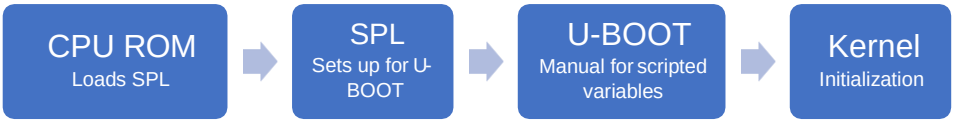

图1. 启动顺序

注意：*i.MX 8M和i.MX 9平台上的顺序略有不同。*

在Arm V8架构上，Arm规定了使用Arm可信固件（ATF）作为启动安全组件的首选方式。ATF首先加载OP-TEE操作系统。OP-TEE操作系统负责初始化安全世界。然后ATF加载即时修改DTB的U-Boot，添加特定节点来加载Linux TEE驱动程序。最终启动Linux操作系统。

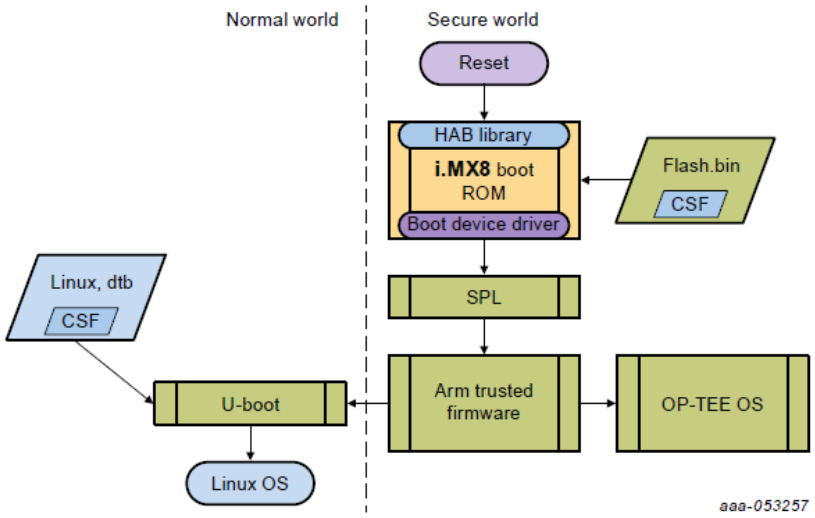

图2. 启动Linux操作系统

4 U-Boot中的Falcon模式

在U-Boot中，引入Falcon模式可加速启动过程，使其无需启动完整U-Boot即可启动Linux内核（或任何映像）。Falcon模式依赖于SPL框架。在大多数实现中，SPL用于启动U-Boot。Falcon模式扩展了这种方式，允许直接从SPL启动Linux内核。

5 从SPL中加载并启动Linux内核

从SPL中加载Linux内核映像时，需要遵循以下步骤：

1. 修改Yocto源代码，将引导分区的大小扩大至50MB。
2. 修改ATF代码，以支持启动带有ATF参数的映像。
3. 在内核中创建falcon `dts`。falcon `dts`的内容包括bootargs和内存信息。
4. 将Linux内核映像与内核Falcon `dtb`构建到`flash.bin`文件中。
5. （可选）使用`uuu`将`flash.bin`文件下载到卡上时，更新U-Boot代码以获取实际引导分区大小。

请查看启动内核映像的蓝色块。

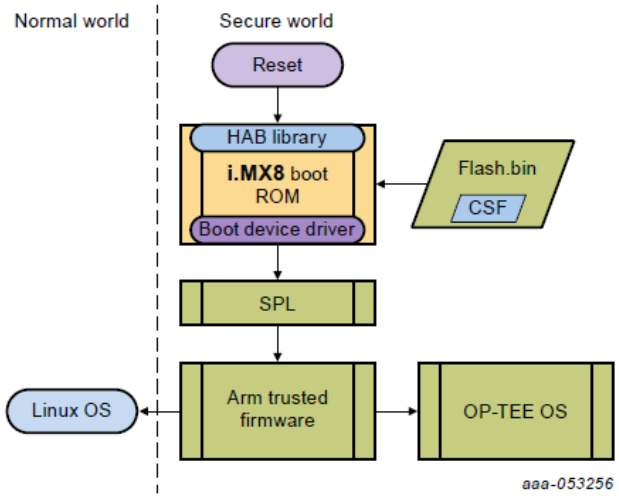

图3. 从SPL中加载并启动Linux内核

5.1 修改Yocto源代码，将引导分区的大小扩大到50MB

在默认版本中，Yocto会为引导分区预留8192kB空间，用于存储MBR/GPT表和`flash.bin`。

将Linux内核映像放入`flash.bin`文件时，8192kB的空间已经不足以容纳30MB内核映像了。因此，我们在Yocto源代码中将引导分区扩大到50MB。

补丁如下：

```
--- ./sources/meta-freescale/wic/imx-imx-boot-bootpart.wks.in.ori
 2023-08-22 15:57:03.817563907 +0800
+++ ./sources/meta-freescale/wic/imx-imx-boot-bootpart.wks.in   2023-05-17
 17:49:43.742919645 +0800
@@ -14,7 +14,8 @@
 #   ${IMX_BOOT_SEEK} 32 or 33kiB, see reference manual
 #
 part u-boot --source rawcopy --sourceparams="file=imx-boot" --ondisk mmcblk --
no-table --align ${IMX_BOOT_SEEK}
-part /boot --source bootimg-partition --ondisk mmcblk --fstype=vfat --label
 boot --active --align 8192 --size 64
+#part /boot --source bootimg-partition --ondisk mmcblk --fstype=vfat --label
 boot --active --align 8192 --size 64
+part /boot --source bootimg-partition --ondisk mmcblk --fstype=vfat --label
 boot --active --align 51200 --size 64
 part / --source rootfs --ondisk mmcblk --fstype=ext4 --label root --align 8192
```

请查看粗体字，将引导分区大小从8192kB更改为51,200kB。

如需了解Yocto中分区命令的更多详情，请参阅https://docs.yoctoproject.org/ref-manual/kickstart.html

5.2 修改ATF 代码，支持在ATF 中使用参数启动映像

在默认版本中，ATF无需任何参数即可启动映像。然而，启动内核并将DTB地址作为参数传递给内核时，就需要修改ATF。

i.MX 8M Nano补丁如下：

```
diff --git a/plat/imx/imx8m/imx8mn/imx8mn_bl31_setup.c b/plat/imx/imx8m/imx8mn/
imx8mn_bl31_setup.c
index c87748a18..0a5a61003 100644
--- a/plat/imx/imx8m/imx8mn/imx8mn_bl31_setup.c
+++ b/plat/imx/imx8m/imx8mn/imx8mn_bl31_setup.c
@@ -244,6 +244,14 @@ void bl31_early_platform_setup2(u_register_t arg0,
 u_register_t arg1,
 #endif
 #endif

+#if defined(USE_FALCON_MODE)
+       /* RAM FDT address */
+       bl33_image_ep_info.args.arg0 = BL32_FDT_ADDR;
+       bl33_image_ep_info.args.arg1 = 0U;
+       bl33_image_ep_info.args.arg2 = 0U;
+       bl33_image_ep_info.args.arg3 = 0U;
+#endif
+
 #if !defined(SPD_opteed) && !defined(SPD_trusty)
        enable_snvs_privileged_access();
```

i.MX 93的补丁如下：

```
diff --git a/plat/imx/imx93/imx93_bl31_setup.c b/plat/imx/imx93/ imx93_bl31_setup.c
index 4c12347cd..1ce739daa 100644
--- a/plat/imx/imx93/imx93_bl31_setup.c
+++ b/plat/imx/imx93/imx93_bl31_setup.c
@@ -80,7 +80,7 @@ void bl31_early_platform_setup2(u_register_t arg0, u_register_t
arg1,
tell BL3-1 where the non-secure software image is located
and the entry state information.
*/
-      bl33_image_ep_info.pc = PLAT_NS_IMAGE_OFFSET;
+      bl33_image_ep_info.pc = (u_register_t)PLAT_NS_IMAGE_OFFSET;
bl33_image_ep_info.spsr = get_spsr_for_bl33_entry();
SET_SECURITY_STATE(bl33_image_ep_info.h.attr, NON_SECURE);

@@ -100,6 +100,15 @@ void bl31_early_platform_setup2(u_register_t arg0,
u_register_t arg1,
bl33_image_ep_info.args.arg3 = BL32_FDT_OVERLAY_ADDR; bl32_image_ep_info.args.arg3
= BL32_FDT_OVERLAY_ADDR;
#endif
+
+#if defined(USE_FALCON_MODE)
+      /* RAM FDT address */
+      bl33_image_ep_info.args.arg0 = (u_register_t)BL32_FDT_ADDR;
+      bl33_image_ep_info.args.arg1 = 0U;
+      bl33_image_ep_info.args.arg2 = 0U;
+      bl33_image_ep_info.args.arg3 = 0U;
+#endif
+
}
```

有关完整补丁，请参阅AN14113SW。

注意：*软件包为i.MX 93提供了类似的`atf`补丁。*

5.3 在内核中创建falcon dts

为了启动内核，不仅要将DTB地址作为参数传递给内核，还要将内核bootargs和内存布局信息传递给内核。这些详细信息通过DTB传递。

启动内核时，请创建falcon dts文件，并包含所需的bootargs、内存布局和其他信息。

以下是Falcon 模式dts文件的模板：

```
// SPDX-License-Identifier: (GPL-2.0+ OR MIT)
/*
 * Copyright 2019 NXP
 */
#include "imx8mn-evk.dts"

/ {
        model = "Falcon Mode Device Tree";
        compatible = "falcon-mode";

        memory {
                reg = <0x00 0x80000000 0x00 0x80000000>;
                device_type = "memory";
        };

        chosen {
                bootargs = "console=ttymxc1,115200 root=/dev/mmcblk1p2 rootwait
 rw";
        };
};
```

dts模板提供了Falcon模式的基本结构，包括内存布局和内核启动参数。你可以根据自己的具体要求进行定制。

下面详细介绍了如何从U-Boot获取dts所需的信息，即bootargs和内存布局。

注意：*通常情况下，内存布局在U-Boot dts中定义并传递给内核，或者直接在内核的dts中定义。这样内核就能知道可用内存的大小。对于大多数平台，我们在内核和U-Boot dts中都有内存布局。大多数平台，例如i.MX 8M、i.MX 8Q，不需要在Falcon dts中定义内存布局。如果内核dts（例如 i.MX93 dts）中没有内存布局信息，那么就在Falcon dts中添加它。*

5.3.1 如何从U-Boot环境中获取bootargs参数

获取bootargs有两种方法：

1. 从<U-Boot>/include/configs/<board name>.h中获取bootargs。以i.MX 8M Nano为例，获取bootargs的步骤如下：
   a. 打开<U-Boot dir>/include/configs/imx8mn_evk.h。
   b. 搜索mmcargs，并查看bootargs是如何分配的。

   ```
   "mmcargs=setenv bootargs ${jh_clk} ${mcore_clk} console=${console} root=
   ${mmcroot}\0 " \
   ```

   c. 尝试获取bootargs。
2. 启动原始U-Boot并获取bootargs。另一种方法是通过运行U-Boot来获得bootargs。
   a. 启动U-Boot，并按任意键停止自动启动。

b. 运行`run mmcargs`。

c. 运行`echo $bootargs`获取bootargs。

```
Hit any key to stop autoboot:  0
u-boot=> run mmcargs
u-boot=> echo $bootargs
console=ttymxc1,115200 root=/dev/mmcblk1p2 rootwait rw
u-boot=>
```

图4. bootargs值

5.3.2 如何获取内存布局

下面介绍内存布局：

```
memory {

    reg = <(baseaddr1) (size1)

    (baseaddr2) (size2)

    ...

    (baseaddrN) (sizeN)>;
    device_type = “memory”;

};
```

其中：

- baseaddrX：已定义内存区的基址。
- sizeX：已定义内存区的大小。
- N：内存区的数量。

这些宏定义了这些值：

- N：如果定义了`PHYS_SDRAM_2_SIZE`，则N为2。否则N为1。
- Baseaddr：`include/configs/imx8mn_evk.h`中的`PHYS_SDRAM`。这是内存中的起始地址。
- 大小：`include/configs/imx8mn_evk.h`中的`PHYS_SDRAM_SIZE`（或`PHYS_SDRAM_2_SIZE`）。这是内存区1（或2）的大小。

注意：*由于i.MX 8M和i.MX 93是64位SoC，因此内存布局地址也是64位。请务必将其与32位地址符号区分开来。*

例如，i.MX 93只有一个内存区。内存起始地址为`0x0000000080000000`，内存大小为`0x0000000080000000`，因此`dts`中的内存布局为：

```
memory {
        reg = <0x00000000 0x80000000 0x00000000 0x80000000>;
         device_type = "memory";
};
```

注意：

- 由于不同平台的DTS文件可能会有很大差异，除了必要的bootargs外，Falcon DTB中的附加配置也不尽相同。

例如，在i.MX 8M平台上，Falcon DTS中只需要包含bootargs，因为内存布局已经在内核DTS中定义。然而，在i.MX 93平台上，Falcon DTS中需要包含内存布局，因为内核DTS中不包含内存布局。编写Falcon模式的DTS时，需要注意这些差异。

- 除了内存布局外，U-Boot在启动内核时可能还会向DTB添加其他内容，如MAC地址、PMIC配置、电路板序列号和U-Boot版本。如果在Falcon模式下启动的内核中发现缺少任何配置，就需要调查这些配置是否包含在Falcon DTB中。在U-Boot代码中，可以在`arch_fixup_fdt`函数（位于`arch/arm/lib/bootm-fdt.c`）中找到修复FDT的代码。

5.4 将Linux内核映像和Falcon DTB编译到flash.bin中

通常，我们使用`imx-mkimage`中的脚本生成`flash.bin`。

`flash.bin`包括映像头、SPL映像和FIT格式的`U-Boot.itb`文件。在此，我们计划将**kernel image**和**`kernel dtb`**添加到`kernel.itb`中，并将`U-Boot.itb`替换为`kernel.itb`。

图5显示了`flash.bin`结构的变化。

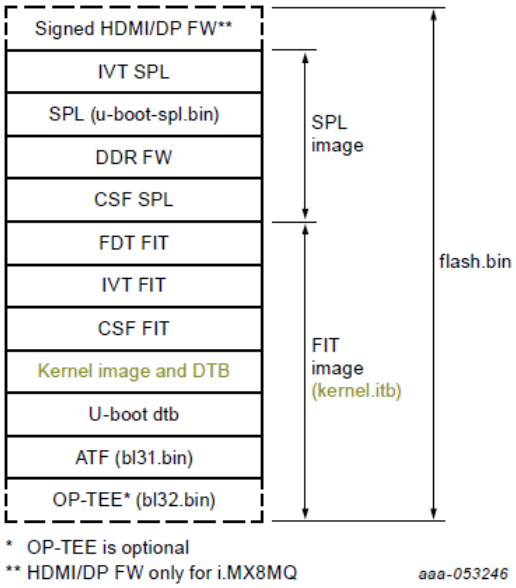

图5. 结构变化

为了生成`U-Boot.itb`，我们需要称为`U-Boot.its`的描述文件。在imx-image中，`U-Boot.its`是使用`mkimage_fit_atf.sh`脚本生成的。因此，如果要包含内核映像和内核DTB的描述段，请修改`mkimage_fit_atf.sh`。

注意：

从`U-Boot.its`，我们可以了解到`U-Boot.itb`不仅包含U-Boot。它还可以包含其他组件，如`bl31.bin` (ATF)、`tee.bin`、`dek_blob`和`.dtb`文件。

图6显示了总图。

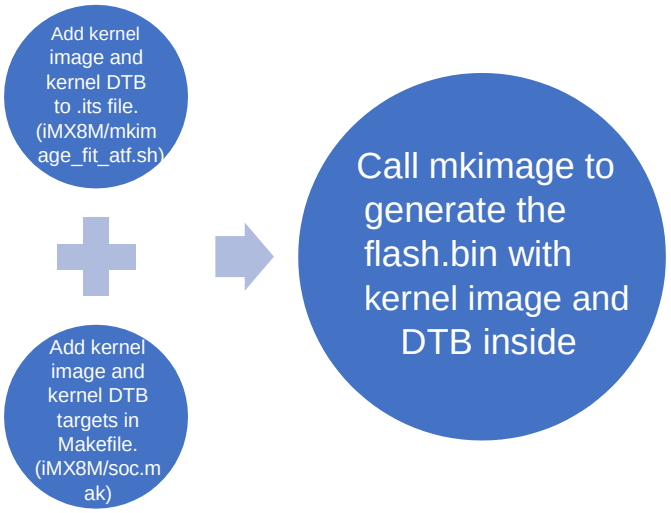

图6. 总图

详情请参见`0001-imx-mkimage-iMX8MN-Add-falcon-mode-support-to-imx-mkimage-o.patch`。

注意：

i.MX 93等其他平台使用不同的映像结构（不是FIT映像，而是i.MX容器）来组织`flash.bin`文件。但操作类似，即用内核映像和设备树blob(DTB)替换`flash.bin`文件中的`u-boot.bin`。

详情请参见`0001-imx-mkimage-iMX93-Add-falcon-mode-support-to-imx-mki.patch`。

5.5 （可选）修改U-Boot可方便使用uuu进行下载

U-Boot快速启动功能中存在一个错误：当下载到SD卡时，引导分区的大小被固定为4MB (0x400000)。当我们的`flash.bin`增加到30MB时，这个大小就不够用了，导致`uuu`下载失败。

```
$ ./uuu.exe -b sd ./imx-boot-imx8mn-lpddr4-evk-sd.bin-flash_evk ./flash_8mn_falcon.bin
uuu (Universal Update Utility) for nxp imx chips -- libuuu_1.4.243-0-ged48c51

Success 0    Failure 1


1:2      4/ 5 [image too large for partition         ] FB: flash bootloader ./flash_8mn_falcon.bin
```

图7. uuu下载失败——uuu日志

```
Detect USB boot. Will enter fastboot mode!
Starting download of 30668272 bytes
.....................................................................
.....................................................................
.....................................................................
...........
downloading of 30668272 bytes finished
Image too large for the partition
```

图8. uuu下载失败——控制台日志

0001-uboot-adjust-bootloader-partition-size-for-uuu-download.patch补丁目的是修复这一问题。

有关此补丁的详细信息，请参阅AN14113SW。

6 测试步骤和结果

测试表明内核能够从SPL启动。

在i.MX 8M Nano和i.MX 93上，使用lf-5.15.71-2.2.0环境构建flash.bin和内核映像，涉及到以下库：

- Repo imx-mkimage
- Repo atf-nxp
- Repo linux-firmware-imx
- Repo uboot-nxp
- Repo linux-lts-nxp

6.1 在i.MX 8M Nano上进行Falcon模式测试

测试步骤和结果如下：

1. 将0001-imx-mkimage-iMX8MN-Add-falcon-mode-support-to-imx-mkimage-o.patch应用于imx-mkimage。
2. 将0001-atf-imx8mn-Add-support-to-start-image-with-parameter.patch补丁应用于atf-nxp。
3. 构建atf并将bl31.bin复制到imx-mkimage。

```
make PLAT=imx8mn bl31
cp build/imx8mn/release/bl31.bin <imx-mkimage>/iMX8M/
```

4. 将0001-linux-imx8mn-Add-falcon-dts.patch补丁应用于linux-lts-nxp。
5. 构建linux内核映像和dtb并复制到imx-mkimage：

```
make ARCH=arm64 imx_v8_defconfig
make ARCH=arm64
cp arch/arm64/boot/Image <imx-mkimage>/iMX8M/kernel_image.bin
cp arch/arm64/boot/dts/freescale/
imx8mn-evk-falcon.dtb <imx-mkimage>/iMX8M/kernel_dtb.bin
```

6. （可选）将0001-uboot-adjust-bootloader-partition-size-for-uuu-download.patch补丁应用于U-Boot。
7. 构建U-Boot。

```
make imx8mn_evk_defconfig
make
```

8. 将mkimage、u-boot-spl.bin和imx8mn-evk.dtb复制到<imx-mkimage/iMX8M>。

```
cp tools/mkimage <imx-mkimage>/iMX8M/mkimage_uboot
cp spl/u-boot-spl.bin <imx-mkimage>/iMX8M/
cp arch/arm/dts/imx8mn-evk.dtb <imx-mkimage>/iMX8M/
```

9. 将lpddr二进制文件从<linux-firmware-imx>/firmware/ddr/synopsys/复制到<imx-mkimage>/iMX8M。

```
cp -rf <linux-firmware-imx>/firmware/ddr/synopsys/*.bin <imx-mkimage>/iMX8M
```

10. 生成flash.bin。

```
make SOC=iMX8MN flash_evk_falcon
```

11. 将flash.bin下载到TF卡的32K偏移值处（或使用uuu下载*flash.bin*）。
12. 启动电路板。

启动日志如下：

```
U-Boot SPL 2022.04-00001-ga0885fceac (Aug 22 2023 - 16:44:59 +0800)
DDRINFO: start DRAM init
DDRINFO: DRAM rate 3200MTS
DDRINFO:ddrphy calibration done
DDRINFO: ddrmix config done
SEC0:  RNG instantiated
Normal Boot
Trying to boot from BOOTROM
Boot Stage: Primary boot
image offset 0x8000, pagesize 0x200, ivt offset 0x0
NOTICE:  BL31: v2.6(release):automotive-13.0.0_1.1.0-rc2-1-g55a839d45
NOTICE:  BL31: Built : 07:48:37, Aug 22 2023
[    0.000000] Booting Linux on physical CPU 0x0000000000 [0x410fd034]
[    0.000000] Linux version 5.15.71-dirty (nxa08304@lsv11258.swis.cn-sha01.nxp.
com) (aarch64-poky-linux-gcc (GCC) 9.2.0, GNU ld (GNU Binutils) 2.32.0.20190204)
 #7 SMP PREEMPT Wed Aug 23 15:23:25 CST 2023
[    0.000000] Machine model: NXP i.MX8MNano EVK board
[    0.000000] efi: UEFI not found.
[    0.000000] Reserved memory: created CMA memory pool at 0x0000000058000000, s
ize 640 MiB
[    0.000000] OF: reserved mem: initialized node linux,cma, compatible id share
d-dma-pool
```

图9. 成功启动的日志

6.2 在i.MX 93上进行Falcon模式测试

测试步骤和结果如下：

1. 将0001-imx-mkimage-iMX93-Add-falcon-mode-support-to-imx-mki.patch补丁应用到imx-mkimage。
2. 将0001-atf-imx93-Add-support-to-start-image-with-parameters.patch补丁应用到atf-nxp。
3. 构建atf并将bl31.bin复制到imx-mkimage。

```
make PLAT=imx93 bl31
cp build/imx93/release/bl31.bin <imx-mkimage>/iMX9/
```

4. 将0002-linux-imx93-Add-falcon-dts.patch补丁应用于linux-lts-nxp。
5. 构建Linux内核映像和dtb，并复制到imx-mkimage：

```
make ARCH=arm64 imx_v8_defconfig
make ARCH=arm64
cp arch/arm64/boot/Image <imx-mkimage>/iMX9/kernel_image.bin
cp arch/arm64/boot/dts/freescale/imx93-11x11-evk-falcon.dtb <imx-mkimage>/
iMX9/kernel_dtb.bin
```

6. （可选）将0001-uboot-adjust-bootloader-partition-size-for-uuu-download.Patch补丁应用于U-Boot。
7. 构建U-Boot。

```
make imx93_11x11_evk_defconfig
make
```

8. 将mkimage和u-boot-spl.bin复制到<imx-mkimage/iMX9>。

```
cp spl/u-boot-spl.bin <imx-mkimage>/iMX9/
cp arch/arm/dts/imx93-11x11-evk.dtb <imx-mkimage>/iMX9/
```

9. 将lpddr二进制文件从<linux-firmware-imx>/firmware/ddr/synopsys/复制到<imx-mkimage>/iMX9。

```
cp -rf <linux-firmware-imx>/firmware/ddr/synopsys/*.bin<imx-mkimage>/iMX9
```

10. 构建flash.bin。

```
make SOC=iMX9 flash_singleboot_falcon
```

11. 将flash.bin下载到TF卡中的32K偏移值处（或使用uuu下载flash.bin）。

启动此电路板。

7 优化

Falcon模式的优点是缩短了启动时间。Falcon模式通过直接启动内核，跳过了U-Boot的启动时间，节省了约3-4秒的U-Boot运行时间。

为了进一步加快启动过程，不仅要优化引导加载程序，还要优化内核和用户空间。

有关内核和用户空间的优化，请参阅《i.MX8M系列Linux启动时间优化》（文件AN13709）中的**第5章**和**第6章**。

8 局限性

我们在本文中实施Falcon模式的解决方案也有一些局限性：

1. eMMC卡使用用户分区启动。由于flash.bin较大，无法将其放入eMMC卡的引导分区。因此，需要使用用户分区启动。
   注意：*另一种方法是使用u-boot命令bootpart-resize来调整引导分区的大小。不过，这个命令可能不适用于eMMC卡。因此，请先咨询eMMC供应商。*
2. 速度。只有使用ROMAPI加载映像时，ROM可支持的eMMC最高速度模式是DDR50（可通过更改启动开关来支持），这一点需要考虑。尽管如此，DDR50的速度为100MB/s，通常足以实现快速启动。

注意：*在《i.MX8M系列的Linux启动时间优化》（文件AN13709）中，还讨论了另一种实现Falcon模式的方法。用户可以查看并选择最适合自己需求的方法。*

9 软件包说明

本应用笔记附带软件包。

表2详细列出了AN14113SW中包含的文件。

表2. 软件包中的文件

文件名	说明	注释
0001-imx-mkimage-iMX8MN-Add-falcon-mode-support-to-imx-mkimage-o.patch	将内核映像和内核DTB作为可加载映像添加到*flash.bin*，并使用i.MX 8M Nano内的内核映像和DTB添加构建选项，以生成*flash.bin*。	基于lf-5. 15.71-2.2.0版本
0001-imx-mkimage-iMX93-Add-falcon-mode-support-to-imx-mki. patch	将内核映像和内核DTB作为可加载映像添加到*flash.bin*，并使用i.MX 93内的内核映像和DTB添加构建选项，以生成*flash.bin*。	基于lf-5. 15.71-2.2.0版本
0001-atf-imx8mn-Add-support-to-start-image-with-parameter.patch	支持*atf*使用i.MX 8M Nano参数启动映像。	基于lf-5. 15.71-2.2.0版本
0001-atf-imx93-Add-support-to-start- image-with-parameters.patch	支持*atf*使用i.MX 93参数启动映像。	基于lf-5. 15.71-2.2.0版本
0001-linux-imx8mn-Add-falcon-dts. patch	将Falcon模式*dts*添加到Linux for i.MX 8M Nano。	基于lf-5. 15.71-2.2.0版本
0002-linux-imx93-Add-falcon-dts. patch	将Falcon模式*dts*添加到Linux for i.MX 93。	基于lf-5. 15.71-2.2.0版本
0001-yocto-change-boot-partition-size-to-50MB.patch	将引导分区扩大到50MB。	基于lf-5. 15.71-2.2.0版本
0001-uboot-adjust-bootloader-partition-size-for-uuu-download.patch	可选补丁用于修复*fastboot*中的引导分区大小。	基于lf-5. 15.71-2.2.0版本

10 参考资料

- 《i.MX移植指南》（文件IMXBSPPG）
- 《i.MX8M系列的Linux启动时间优化》（文件AN13709）
- 了解U-Boot Falcon模式

11 关于本文中源代码的说明

本文中所示的示例代码具有以下版权和BSD-3-Clause许可：
2024年恩智浦版权所有；在满足以下条件的情况下，可以源代码和二进制文件的形式重新分发和使用本源代码（无论是否经过修改）：

1. 重新分发源代码必须保留上述版权声明、这些条件和以下免责声明。
2. 以二进制文件形式重新分发时，必须在文档和/或随分发提供的其他材料中复制上述版权声明、这些条件和以下免责声明。
3. 未经事先书面许可，不得使用版权所有者的姓名或参与者的姓名为本软件的衍生产品进行背书或推广。

本软件由版权所有者和参与者“按原样”提供，不承担任何明示或暗示的担保责任，包括但不限于对适销性和特定用途适用性的暗示保证。在任何情况下，无论因何种原因或根据何种法律条例，版权所有者或参与者均不对因使用本软件而导致的任何直接、间接、偶然、特殊、惩戒性或后果性损害（包括但不限于采购替代商品或服务；使用损失、数据损失或利润损失或业务中断）承担责任，无论是因合同、严格责任还是侵权行为（包括疏忽或其他原因）造成的，即使事先被告知有此类损害的可能性也不例外。

12 修订历史

表3. 修订历史

文档ID	发布日期	说明
AN14113 v.2	2024年2月22日	第6节和第8节已更新。
AN14113 v.1	2023年11月13日	初始版本

Legal information

Definitions

Draft — A draft status on a document indicates that the content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included in a draft version of a document and shall have no liability for the consequences of use of such information.

Disclaimers

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information. NXP Semiconductors takes no responsibility for the content in this document if provided by an information source outside of NXP Semiconductors.

In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory.

Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of NXP Semiconductors.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors and its suppliers accept no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

Terms and conditions of commercial sale — NXP Semiconductors products are sold subject to the general terms and conditions of commercial sale, as published at https://www.nxp.com.cn/profile/terms, unless otherwise agreed in a valid written individual agreement. In case an individual agreement is concluded only the terms and conditions of the respective agreement shall apply. NXP Semiconductors hereby expressly objects to applying the customer's general terms and conditions with regard to the purchase of NXP Semiconductors products by customer.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

Suitability for use in non-automotive qualified products — Unless this document expressly states that this specific NXP Semiconductors product is automotive qualified, the product is not suitable for automotive use. It is neither qualified nor tested in accordance with automotive testing or application requirements. NXP Semiconductors accepts no liability for inclusion and/or use of non-automotive qualified products in automotive equipment or applications.

In the event that customer uses the product for design-in and use in automotive applications to automotive specifications and standards, customer (a) shall use the product without NXP Semiconductors' warranty of the product for such automotive applications, use and specifications, and (b) whenever customer uses the product for automotive applications beyond NXP Semiconductors' specifications such use shall be solely at customer's own risk, and (c) customer fully indemnifies NXP Semiconductors for any liability, damages or failed product claims resulting from customer design and use of the product for automotive applications beyond NXP Semiconductors' standard warranty and NXP Semiconductors' product specifications.

Translations — A non-English (translated) version of a document, including the legal information in that document, is for reference only. The English version shall prevail in case of any discrepancy between the translated and English versions.

Security — Customer understands that all NXP products may be subject to unidentified vulnerabilities or may support established security standards or specifications with known limitations. Customer is responsible for the design and operation of its applications and products throughout their lifecycles to reduce the effect of these vulnerabilities on customer's applications and products. Customer's responsibility also extends to other open and/or proprietary technologies supported by NXP products for use in customer's applications. NXP accepts no liability for any vulnerability. Customer should regularly check security updates from NXP and follow up appropriately.

Customer shall select products with security features that best meet rules, regulations, and standards of the intended application and make the ultimate design decisions regarding its products and is solely responsible for compliance with all legal, regulatory, and security related requirements concerning its products, regardless of any information or support that may be provided by NXP.

NXP has a Product Security Incident Response Team (PSIRT) (reachable at PSIRT@nxp.com) that manages the investigation, reporting, and solution release to security vulnerabilities of NXP products.

NXP B.V. — NXP B.V. is not an operating company and it does not distribute or sell products.

Trademarks

Notice: All referenced brands, product names, service names, and trademarks are the property of their respective owners.

NXP — wordmark and logo are trademarks of NXP B.V.

目录

For more information, please visit: https://www.nxp.com.cn

Date of release: 22 February 2024
Document identifier: AN14113