storage

https://github.com/BPI-SINOVOIP/BPI-R64-BSP/blob/master/scripts/dd_download.sh

image: https://github.com/BPI-SINOVOIP/BPI-R64-BSP/blob/master/scripts/bootloader.sh

new way

With the opensource ATF we switched to full 64bit mode,fip (bl31+uboot) and fit kernel

Details on my ATF repo: https://github.com/frank-w/BPI-R64-ATF

```
sudo dd of=$target if=./fip_sdmmc.bin bs=512 seek=2048
```

legacy way

SD

```
PRELOADER=$TOPDIR/mt-pack/mtk/${TARGET_PRODUCT}/bin/preloader_bpi-
r64_forsdcard-2k.img
ATF=$TOPDIR/mt-pack/mtk/${TARGET_PRODUCT}/bin/BPI-R64-atf.img
UBOOT=$TOPDIR/u-boot-mt/u-boot-mtk.bin

HEAD0=$TOPDIR/mt-pack/mtk/${TARGET_PRODUCT}/bin/BPI-R2-HEAD440-0k.img
HEAD1=$TOPDIR/mt-pack/mtk/${TARGET_PRODUCT}/bin/BPI-R2-HEAD1-512b.img

sudo dd if=$HEAD0   of=$O bs=512 seek=0 #0
sudo dd if=$HEAD1   of=$O bs=512 seek=1 #512 = 0x200

sudo dd if=$PRELOADER   of=$O bs=1k seek=2 #2k = 0x800

sudo dd if=$ATF     of=$O bs=1k seek=512 #512k = 0x80000

sudo dd if=$UBOOT   of=$O bs=1k seek=768 #768k = 0xC0000

# partition1 /dev/sdc1 vfat 204800~327679
# partition2 /dev/sdc2 ext4 327680~end

root@x:~# sfdisk /dev/sdb < r64_parttable.dat
root@x:~# mkfs.vfat /dev/sdb1
root@x:~# mkfs.ext4 /dev/sdb2
root@x:~# fatlabel /dev/sdb1 BPI-BOOT
root@x:~# e2label /dev/sdb2 BPI-ROOT
```

https://github.com/BPI-SINOVOIP/BPI-R64-BSP/tree/master/mt-pack/mtk/bpi-r64/bin

only uboot-mtk.bin (with littlekernel=LK) works

A better ATF is mt7622_atf_push_wps_uboot_64.img (can be found on my gdrive too) supports booting 32bit uboot by default and 64bit uboot by pressing wps-button

http://forum.banana-pi.org/t/bpi-r64-current-u-boot-support/10077/79

EMMC

this Preloader have to be used: preloader_evb7622_64_foremmc.bin (copy on my gdrive)

- boot from sd-card

```
root@bpi-r64:~# ls /dev/mmcblk*
/dev/mmcblk0    /dev/mmcblk0p2  /dev/mmcblk1boot0  /dev/mmcblk1rpmb
/dev/mmcblk0p1  /dev/mmcblk1    /dev/mmcblk1boot1
#configure lan-port
root@bpi-r64:~# ip addr add 192.168.0.18/24 dev eth0
root@bpi-r64:~# ip route add default via 192.168.0.10
root@bpi-r64:~# echo "nameserver 192.168.0.10">/etc/resolv.conf
#download preloader and flash it
root@bpi-r64:~# wget
https://github.com/BPI-SINOVOIP/BPI-R64-bsp-4.19/raw/master/mt-pack/mtk/bpi-r64/configs/default/linux-4.19/preloader_evb7622_64_foremmc.bin
root@bpi-r64:~# echo 0 > /sys/block/mmcblk1boot0/force_ro
root@bpi-r64:~# dd if=preloader_evb7622_64_foremmc.bin of=/dev/mmcblk1boot0
```

- boot0-block exists, but mmc-utils showing partconfig 0x0 (should be 0x48), emmc-boot hangs at bootrom:

```
F0: 102B 0000
F5: 480A 0031
F5: 480A 0031
F3: 4000 0036
F2: 300C 0000
00: 1005 0000
F5: 480A 0031
F5: 480A 0031
F3: 4000 0036
F2: 300C 0000
01: 102A 0001
02: 1005 0000
BP: 0000 00C0 [0001]
T0: 0000 035F [000F]
System halt!
```

```
root@bpi-r64:~# ./mmc extcsd read /dev/mmcblk1 | grep 'PARTITION_CONFIG'
Boot configuration bytes [PARTITION_CONFIG: 0x00]
```

```
root@bpi-r64:~# ./mmc bootpart enable 1 1 /dev/mmcblk1
root@bpi-r64:~# ./mmc extcsd read /dev/mmcblk1 | grep 'PARTITION_CONFIG'
Boot configuration bytes [PARTITION_CONFIG: 0x48]
```

This can also be done in recent uboot

```
mmc partconf 0
mmc partconf 0 1 1 0
```

after this steps preloader gets loaded (i had only flashed preloader here and nothing alse, so error is correct...full boot needs additional headers HEAD0,HEAD1,SD-Preloader,ATF and UBOOT)...

```
F0: 102B 0000
F5: 480A 0031
F5: 480A 0031
F3: 0000 0000
V0: 0000 0000 [0001]
00: 0000 0000
BP: 0000 0041 [0000]
G0: 0190 0000
T0: 0000 039F [000F]
Jump to BL
UNIVPLL_CON0 = 0xFE000000!!!
mt_pll_init: Set pll frequency for 25M crystal
RAM_CONSOLE preloader last status: 0x0 0x0 0x0 0x0 0x0 0x0
[PMIC_WRAP]wrap_init pass,the return value=0.
[pmic_init] Preloader Start.................
[pmic_init] MT6380 CHIP Code, reg_val = 0, 1:E2  0:E3
[pmic_init] Done...................
Chip part number:7622A
MT7622 Version: 1.2.7, (iPA)
SSC OFF
mt_pll_post_init: mt_get_cpu_freq = 1350000Khz
mt_pll_post_init: mt_get_mem_freq = 1600000Khz
mt_pll_post_init: mt_get_bus_freq = 1119920Khz
[PLFM] Init I2C: OK(0)
[BLDR] Build Time: 20190927-141930
==== Dump RGU Reg ========
RGU MODE:     4D
RGU LENGTH:   FFE0
RGU STA:      0
RGU INTERVAL: FFF
RGU SWSYSRST: 8000
==== Dump RGU Reg End ====
RGU: g_rgu_satus:0
 mtk_wdt_mode_config  mode value=10, tmp:22000010
PL P ON
WDT does not trigger reboot
WDT NONRST=0x20000000
WDT IRQ_EN=0x340003
```

```
RGU mtk_wdt_init:MTK_WDT_DEBUG_CTL(590200F3)
[EMI] MDL number = 2
[EMI] DRAMC calibration start
[EMI] DRAMC calibration end
[EMI]rank0 size: 0x40000000
[MEM] complex R/W mem test pass
RAM_CONSOLE wdt status (0x0)=0x0
[mmc_init]: msdc0 start mmc_init_host() in PL...
[msdc_init]: msdc0 Host controller intialization start
[SD0] Pins mode(1), none(0), down(1), up(2), keep(3)
[SD0] Pins mode(2), none(0), down(1), up(2), keep(3)
[info][msdc_set_startbit 1127] read data start bit at rising edge
[info][msdc_config_clksrc] input clock is 400000kHz
[SD0] Bus Width: 1
[info][msdc_config_clksrc] input clock is 400000kHz
[info][msdc_set_startbit 1127] read data start bit at rising edge
[SD0] SET_CLK(260kHz): SCLK(259kHz) MODE(0) DDR(0) DIV(385) DS(0) RS(0)
[msdc_init]: msdc0 Host controller intialization done
[mmc_init]: msdc0 start mmc_init_card() in PL...
[mmc_init_card]: start
[info][msdc_config_clksrc] input clock is 400000kHz
[info][msdc_set_startbit 1127] read data start bit at rising edge
[SD0] SET_CLK(260kHz): SCLK(259kHz) MODE(0) DDR(0) DIV(385) DS(0) RS(0)
[SD0] Bus Width: 8
[SD0] Switch to High-Speed mode!
[info][msdc_config_clksrc] input clock is 400000kHz
[info][msdc_set_startbit 1127] read data start bit at rising edge
[SD0] SET_CLK(260kHz): SCLK(259kHz) MODE(2) DDR(1) DIV(192) DS(0) RS(0)
[SD0] Bus Width: 8
[SD0] Size: 7456 MB, Max.Speed: 52000 kHz, blklen(512), nblks(15269888),
ro(0)
[mmc_init_mem_card 3140][SD0] Initialized, eMMC50
before host->cur_bus_clk(259740)
[info][msdc_config_clksrc] input clock is 400000kHz
[info][msdc_set_startbit 1127] read data start bit at rising edge
[SD0] SET_CLK(52000kHz): SCLK(50000kHz) MODE(2) DDR(1) DIV(1) DS(0) RS(0)
host->cur_bus_clk(50000000)
[mmc_init_card]: finish successfully
[PLFM] Init Boot Device: OK(0)
[GPT_PL](BPI)Parsing Primary GPT now...
[GPT_PL]check header, err(signature 0x0000000000000000!=0x5452415020494645)
[GPT_PL]Success to find valid GPT.
[PART] blksz: 512B
[PART] [0x0000000000020000-0x000000000007FFFF] "preloader" (768 blocks)
[PART] [0x0000000000080000-0x00000000000BFFFF] "tee1" (512 blocks)
[PART] [0x00000000000C0000-0x000000000013FFFF] "lk" (1024 blocks)
Device APC domain init setup:
Domain Setup (0x0)
Domain Setup (0x0)
Device APC domain after setup:
Domain Setup (0x0)
```

```
Domain Setup (0x0)
[get_part] part->nr_sects=768, part->info->name=preloader
[get_part] part->nr_sects=512, part->info->name=tee1
[get_part] part->nr_sects=1024, part->info->name=lk
load lk (ret=-1)
[BLDR] Second Bootloader Load Failed
PL fatal error...
```

maybe write partitiontable

```
root@bpi-r64:~# sfdisk -d /dev/mmcblk0 > r64_parttable.dat
root@bpi-r64:~# sfdisk /dev/mmcblk1 < r64_parttable.dat
...
root@bpi-r64:~# ls /dev/mmcblk*
/dev/mmcblk0    /dev/mmcblk1       /dev/mmcblk1p1
/dev/mmcblk0p1  /dev/mmcblk1boot0  /dev/mmcblk1p2
/dev/mmcblk0p2  /dev/mmcblk1boot1  /dev/mmcblk1rpmb
```

boot from emmc till uboot: http://forum.banana-pi.org/t/bpi-r64-quick-start-boot-from-emmc/9809/36

i've created a small script with all headers needed for emmc-boot:

https://drive.google.com/open?id=1a8yCzwiCAcqtHHXDLhRzMdItsMC7ndJr this erases content of emmc! if you do not want this, comment out lines for writing partition table and formatting