U-Boot

Mtk uboot: https://github.com/mtk-openwrt/u-boot

Mtk atf: https://github.com/mtk-openwrt/arm-trusted-firmware

My uboot: https://github.com/frank-w/u-boot/tree/2022-10-r3

building

Uboot needs to build into atf bl33.

- https://github.com/frank-w/u-boot/tree/2022-10-r3
- https://github.com/frank-w/u-boot/tree/r3-atf

So basicly do these in both branches (first uboot then atf)

```
./build.sh importconfig
./build.sh
```

Then flash build/mt7986/release/ bl2.img to bl2-partition and fip.bin to fip-partition

```
sudo dd if=build/mt7986/release/bl2.img of=/dev/sdb1
sudo dd if=build/mt7986/release/fip.bin of=/dev/sdb4
```

ATF branch contains gpt file for flashing first and script to create basic image with bootchain

booting kernel

```
setenv kaddr 0x46000000
ls mmc 0:5
fatload mmc 0:5 ${kaddr} bpi-r3-5.19.0-rc1-r3.itb
#setenv bootargs "earlycon=uart8250,mmio32,0x11002000 console=ttyS0"
bootm ${kaddr}
```

board seems to hang in 5.19-rc1 when console=ttyS0 is passed

with my environment it is easy to boot kernel from usb

```
BPI-R3> usb start
BPI-R3> setenv device usb #was mmc before
BPI-R3> setenv partition 0:1 #was 0:5 before for mmc
BPI-R3> setenv fit bpi-r3-sfp.itb #change filename if needed
BPI-R3> run newboot #will load bpi-r3.itb
```

load fip with alternate dtb (like from my current 5.19-r3 branch, second config contains emmc dtb)

```
BPI-R3> fatload usb 0:1 $loadaddr bpi-r3.itb
17032308 bytes read in 1427 ms (11.4 MiB/s)
BPI-R3> bootm ${loadaddr}#conf-2
```

load external initrd (run before „run newboot“):

```
BPI-R3> setenv rdaddr 0x48000000
BPI-R3> setenv root "/dev/ram0 rw"
BPI-R3> setenv bootopts "debug=7 initrd=${rdaddr},20M"
BPI-R3> fatload usb 0:1 ${rdaddr} rootfs.cpio.zst
```

latest uboot with usb, initrd and bootconf

```
BPI-R3> run useusb
BPI-R3> setenv initrd rootfs.cpio.zst
BPI-R3> setenv fit bpi-r3-sfp-no-rd.itb
BPI-R3> setenv bootconf "#conf-2" #
BPI-R3> run newboot
```

```
#using spi dt-overlay
BPI-R3> setenv bootconf "#conf-sd-nor" #conf-sd-nor=sd base dts + nor dt
overlay
```

```
setenv buildargs 'setenv bootargs "board=${board}
earlycon=uart8250,mmio32,0x11002000  ${bootopts} root=${root}"'
```

bootmedium selection

	top		bottom	
	sw1/A	sw2/B (spi/mmc)	sw5/C (nor/nand)	sw6/D (sd/emmc)
spim-nor	low	low	low	x
spim-nand	high	low	high	x
emmc	low	high	x	low
sd	high	high	x	high

To access emmc you need to go over spi-nand/nor:

Bootup with sdcard (put spi-nand bl2 + fip with emmc-and usb-support as files on card) and flash them like described here then boot from spi (sw2 to boot from spi-nand + change sw6 to have access to emmc) and then flash emmc. Finally change sw2 back to boot from emmc and sw1 has to be changed too.

nand flash

```
BPI-R3> ls mmc 0:5
 17019020   bpi-r3.itb
   579401   fip_nand.bin
```

```
   217712   bl2_nand.img

3 file(s), 0 dir(s)

BPI-R3> mtd list
List of MTD devices:
* spi-nand0
  - device: spi_nand@1
  - parent: spi@1100a000
  - driver: spi_nand
  - path: /spi@1100a000/spi_nand@1
  - type: NAND flash
  - block size: 0x20000 bytes
  - min I/O: 0x800 bytes
  - OOB size: 64 bytes
  - OOB available: 24 bytes
  - 0x000000000000-0x000008000000 : "spi-nand0"
* nmbm0
  - type: Unknown
  - block size: 0x20000 bytes
  - min I/O: 0x800 bytes
  - OOB size: 64 bytes
  - OOB available: 24 bytes
  - 0x000000000000-0x000007800000 : "nmbm0"
          - 0x000000000000-0x000000100000 : "bl2"
          - 0x000000100000-0x000000180000 : "u-boot-env"
          - 0x000000180000-0x000000380000 : "factory"
          - 0x000000380000-0x000000580000 : "fip"
          - 0x000000580000-0x000004580000 : "ubi"

BPI-R3> printenv loadaddr
loadaddr=0x46000000

BPI-R3> mtd erase spi-nand0
Erasing 0x00000000 ... 0x07ffffff (1024 eraseblock(s))
BPI-R3> fatload mmc 0:5 $loadaddr bl2_nand.img
217712 bytes read in 24 ms (8.7 MiB/s)
BPI-R3> mtd write spi-nand0 $loadaddr 0x0 0x100000
Writing 1048576 byte(s) (512 page(s)) at offset 0x00000000

BPI-R3> fatload mmc 0:5 $loadaddr fip_nand.bin
579401 bytes read in 60 ms (9.2 MiB/s)
BPI-R3> mtd write spi-nand0 $loadaddr 0x380000 0x200000
Writing 2097152 byte(s) (1024 page(s)) at offset 0x00380000
```

usb

```
BPI-R3> usb start
starting USB...
```

```
Bus xhci@11200000: xhci-mtk xhci@11200000: hcd: 0x0000000011200000, ippc:
0x0000
xhci-mtk xhci@11200000: ports disabled mask: u3p-0x0, u2p-0x0
xhci-mtk xhci@11200000: u2p:2, u3p:1
Register 300010f NbrPorts 3
Starting the controller
USB XHCI 1.10
scanning bus xhci@11200000 for devices... 3 USB Device(s) found
       scanning usb for storage devices... 1 Storage Device(s) found
BPI-R3> ls usb 0:1
            efi/
  1023582   ramdisk.img
  1365903   initrd.img
  2956137   install.img
 456908800   system.sfs
  4767728   kernel

5 file(s), 1 dir(s)

BPI-R3>
```

from testing usb-sockets near gpio-header (connected to an usb2-hub):

```
BPI-R3> usb tree
USB device tree:
  1  Hub (5 Gb/s, 0mA)
  |  U-Boot XHCI Host Controller
  |
  +-2  Hub (480 Mb/s, 100mA)
    |   USB 2.0 Hub
    |
    +-3  Mass Storage (480 Mb/s, 200mA)
         USB      Flash Disk       906B030002F4
```

writing emmc

get emmc/gpt information:

```
BPI-R3> mmc info
Device: mmc@11230000
Manufacturer ID: 11
OEM: 0
Name: 008GB0
Bus Speed: 52000000
Mode: MMC High Speed (52MHz)
Rd Block Len: 512
MMC version 5.1
High Capacity: Yes
```

```
Capacity: 7.3 GiB
Bus Width: 8-bit
Erase Group Size: 512 KiB
HC WP Group Size: 4 MiB
User Capacity: 7.3 GiB WRREL
Boot Capacity: 4 MiB ENH
RPMB Capacity: 4 MiB ENH
Boot area 0 is not write protected
Boot area 1 is not write protected

BPI-R3> gpt read mmc 0
Partition 1:
Start 0MiB, size 0MiB
Block size 512, name gpt
Type U-Boot, bootable 0
UUID df7a38f6-9df4-11ec-a1e0-1c1b0d6d28f5

Partition 2:
Start 4MiB, size 0MiB
Block size 512, name u-boot-env
Type U-Boot, bootable 0
UUID 19a4763a-6b19-4a4b-a0c4-8cc34f4c2ab9

Partition 3:
Start 4MiB, size 2MiB
Block size 512, name factory
Type U-Boot, bootable 0
UUID 8142c1b2-1697-41d9-b1bf-a88d76c7213f

Partition 4:
Start 6MiB, size 2MiB
Block size 512, name fip
Type U-Boot, bootable 0
UUID 18de6587-4f17-4e08-a6c9-d9d3d424f4c5

Partition 5:
Start 8MiB, size 32MiB
Block size 512, name kernel
Type U-Boot, bootable 0
UUID 971f7556-ef1a-44cd-8b28-0cf8100b9c7e

Partition 6:
Start 40MiB, size 256MiB
Block size 512, name rootfs
Type U-Boot, bootable 0
UUID 309a3e76-270b-41b2-b5d5-ed8154e7542b

success!
```

load gpt file and flash it

```
BPI-R3> usb start
starting USB...
Bus xhci@11200000: xhci-mtk xhci@11200000: hcd: 0x0000000011200000, ippc:
0x0000
xhci-mtk xhci@11200000: ports disabled mask: u3p-0x0, u2p-0x0
xhci-mtk xhci@11200000: u2p:2, u3p:1
Register 300010f NbrPorts 3
Starting the controller
USB XHCI 1.10
scanning bus xhci@11200000 for devices... 3 USB Device(s) found
       scanning usb for storage devices... 1 Storage Device(s) found
BPI-R3> ls usb 0:1
            efi/
  1365903   initrd.img
  2956137   install.img
  4767728   kernel
   606240   uboot_mtk_emmc_usb.bin
   217712   bl2_snand_usb.img
   631937   fip_snand_usb.bin
   634345   fip_emmc_usb.bin
   601344   uboot_emmc_usb.bin
    17408   gpt_emmc_100m6g.img
   200072   bl2_emmc_usb.img
 17032308   bpi-r3.itb

11 file(s), 1 dir(s)

BPI-R3> fatload usb 0:1 $loadaddr gpt_emmc_100m6g.img
17408 bytes read in 4 ms (4.2 MiB/s)
BPI-R3> mmc erase 0x0 0x400;mmc write ${loadaddr} 0x0 0x400

MMC erase: dev # 0, block # 0, count 1024 ... 1024 blocks erased: OK

MMC write: dev # 0, block # 0, count 1024 ... 1024 blocks written: OK
```

verify gpt is changed (kernel is now 100MiB - 32 before, rootfs 6144MiB - 256 before):

```
BPI-R3> gpt read mmc 0
Partition 1:
Start 0MiB, size 0MiB
Block size 512, name gpt
Type U-Boot, bootable 0
UUID 4a32da88-f334-11ec-94f0-112beaedcdcb

Partition 2:
Start 4MiB, size 0MiB
Block size 512, name u-boot-env
Type U-Boot, bootable 0
UUID 19a4763a-6b19-4a4b-a0c4-8cc34f4c2ab9
```

```
Partition 3:
Start 4MiB, size 2MiB
Block size 512, name factory
Type U-Boot, bootable 0
UUID 8142c1b2-1697-41d9-b1bf-a88d76c7213f

Partition 4:
Start 6MiB, size 2MiB
Block size 512, name fip
Type U-Boot, bootable 0
UUID 18de6587-4f17-4e08-a6c9-d9d3d424f4c5

Partition 5:
Start 8MiB, size 100MiB
Block size 512, name kernel
Type U-Boot, bootable 0
UUID 971f7556-ef1a-44cd-8b28-0cf8100b9c7e

Partition 6:
Start 108MiB, size 6144MiB
Block size 512, name rootfs
Type U-Boot, bootable 0
UUID 309a3e76-270b-41b2-b5d5-ed8154e7542b

success!
BPI-R3>
```

now flash bl2 to boot0 block and fip to fip partition

```
BPI-R3> mmc dev 0
switch to partitions #0, OK
mmc0(part 0) is current device
#switch to boot0 block
BPI-R3> mmc partconf 0 1 1 1
BPI-R3> mmc erase 0x0 0x400
BPI-R3> fatload usb 0:1 $loadaddr bl2_emmc_usb.img
200072 bytes read in 19 ms (10 MiB/s)
BPI-R3> mmc write ${loadaddr} 0x0 0x400

MMC write: dev # 0, block # 0, count 1024 ... 1024 blocks written: OK
#switch back to user-partition
BPI-R3> mmc partconf 0 1 1 0

#fip partition starts at blockoffset 0x3400 (start-value from json 13312 in
hex)
BPI-R3> mmc erase 0x3400 0x1000

MMC erase: dev # 0, block # 13312, count 4096 ... 4096 blocks erased: OK
BPI-R3> fatload usb 0:1 $loadaddr fip_emmc_usb.bin
634345 bytes read in 55 ms (11 MiB/s)
```

```
BPI-R3> mmc write ${loadaddr} 0x3400 0x1000

MMC write: dev # 0, block # 13312, count 4096 ... 4096 blocks written: OK
BPI-R3>
```

If boot from emmc does not work,maybe bootbus needs to be changed to 0:

```
mmc bootbus 0 0 0 0
```

chainload uboot

u-boot.bin (without mtk header and not fip) can be chainloaded from another uboot for testing uboot-features without flashing

```
BPI-R3> fatload usb 0:1 $loadaddr uboot_emmc_usb.bin
601344 bytes read in 52 ms (11 MiB/s)
BPI-R3> go $loadaddr
## Starting application at 0x46000000 ...

U-Boot 2022.04-00557-g726479add2f1 (Jul 05 2022 - 18:14:31 +0200)
```

From:
https://wiki.fw-web.de/ - **FW-WEB Wiki**

Permanent link:
https://wiki.fw-web.de/doku.php?id=en:bpi-r3:uboot&rev=1665378681

Last update: **2023/06/08 17:06**