Tornando mais fácil para o u-boot encontrar uImage e uinitrd

Eu costumo mexer com o teste de imagens de raiz, kernels e initrd em meus sistemas baseados em kirkwood usando flash sticks usb. Descobri que, embora Jeff tenha feito um trabalho brilhante no uboot, na época em que comecei a mexer com imagens de inicialização personalizadas no meu dosckstar, seu ambiente padrão não pôde inicializar diretamente todas as minhas imagens de teste que estavam mudando rapidamente no layout. Para contornar isso, comecei a fazer alterações em seu ambiente.

Eu criei um ambiente personalizado que era mais flexível e permiti que eu inicializasse a partir de qualquer porta usb, desde que as seguintes limitações sejam atendidas:

- O kernel deve ser chamado de uImage ou deve estar vinculado a esse nome
- O initrd deve ser chamado de uinitrd ou deve estar vinculado a esse nome
- uImage e uinitrd devem estar contidos nos primeiros 4 dispositivos usb detectados
- uImage e uinitrd devem estar na raiz ou em um subdiretório de inicialização no dispositivo que os mantém
- O sistema de arquivos root será rotulado como “root” e possivelmente deve estar formatado em ext3

Aqui está o ambiente despejado de fw_printenv: (extraído do meu GoFlexNet)

```
arcNumber=<put here your device's correct arcNumber>
baudrate=115200
boot_flash_kernel=nand read $flash_kernel_load_addr $flash_kernel_offest
$flash_kernel_size; bootm $flash_kernel_load_addr
bootargs=console=ttyS0,115200 mtdparts=orion_nand:1M(u-
boot),4M(uImage),32M(rootfs),-(data) root=/dev/sda2 ro rootfstype=ext2
bootcmd=run usb_scan; run set_flash_bootargs; run boot_flash_kernel
bootdelay=3
console=console=ttyS0,115200
ethact=egiga0
ethaddr=<put here your mac address>
flash_kernel_load_addr=0x6400000
flash_kernel_offest=0x100000
flash_kernel_size=0x300000
flash_root_fs=root=/dev/mtdblock2
flash_root_fstype=rootfstype=jffs2
led_error=orange blinking
led_exit=green off
led_init=green blinking
mainlineLinux=yes
mtdids=nand0=orion_nand
mtdparts=mtdparts=orion_nand:1M(u-boot),4M(uImage),32M(rootfs),-(data)
partition=nand0,2
root_fs=root=LABEL=root
rootfstype=rootfs=ext3
sb_scan_1=usb=0:2 dev=sda2
```

```
set_flash_bootargs=setenv bootargs ${console} ${mtdparts} ${flash_root_fs}
ro ${flash_root_fstype}
set_usb_bootargs=setenv bootargs ${console} ${mtdparts} ${root_fs} ro
${root_fstype}
stderr=serial
stdin=serial
stdout=serial
usb_boot_6=setenv root_fs root=/dev/sdg${usb_dev_part}
usb_dev_list=3 2 1 0
usb_dev_part=1
usb_part_list=4 3 2 1
usb_scan=usb start; setenv usb_boot_dev none; for dev in  $usb_dev_list; do
test $dev -eq 0 && setenv devname /dev/sda ; test $dev -eq 1 &&  setenv
devname /dev/sdb ; test $dev -eq 2 && setenv devname /dev/sdc ; test $dev -
eq 3 && setenv devname /dev/sdd ; echo $devname ; for part in
$usb_part_list; do echo $dev $part ; if ext2load usb ${dev}:$part 0x800000
/boot/uImage 10 ; then setenv usb_boot_dev $dev:$part ; setenv usb_boot_dir
/boot ; fi  ; if ext2load usb ${dev}:$part 0x800000 /uImage 10 ; then setenv
usb_boot_dev $dev:$part ; setenv usb_boot_dir ; fi  ; done; done; if test
"$usb_boot_dev" = "none" ; then echo "No USB bootable device found" ; else
echo "USB device $usb_boot_dev is bootable" ; setenv bootargs $console
$mtdparts $root_fs ro $rootfstype ; echo $bootargs ; sleep 1; ext2load usb
$usb_boot_dev 0x800000 $usb_boot_dir/uImage && setenv usb_boot_address
0x800000 ; ext2load usb $usb_boot_dev 0x1100000 $usb_boot_dir/uinitrd &&
setenv usb_boot_address 0x800000 0x1100000 ; bootm $usb_boot_address ; fi;
```

A rotulagem do sistema de arquivos root foi uma solução alternativa quando comecei a receber problemas nas alterações de nomenclatura de dispositivos quando comecei a usar as unidades SATA para armazenamento NAS. Na prática, o que estava acontecendo era que o uboot pensava que o root estava em sda, mas uma vez que o kernel inicializasse, ele detectaria as unidades SATA antes disso e nomeava o memory stick usb após a última unidade SATA, causando um kernel panic, pois o initrd tenta montar o sistema de arquivos raiz. Assim, alterei o ambiente de inicialização para passar o dispositivo raiz como um rótulo em vez de um caminho de dispositivo. Isso permite que eu inicialize meu GoFlexNet corretamente de qualquer pen drive (mesmo através de um hub usb), independentemente da presença das unidades SATA.

Pelo que entendi, os últimos uboots realmente fazem mais do que isso agora... então aqui está meu último ambiente híbrido:

```
arcNumber=3089
baudrate=115200
bifsload=0x800000 /boot/$kernel && ubifsload 0x1100000 /boot/$initrd; then
bootm 0x800000 0x1100000; fi
bootargs=${console} ${mtdparts} ${root_fs} ro ${rootfstype}
bootcmd=usb start; run force_rescue_bootcmd; run ubifs_bootcmd; run
usb_scan; usb stop; run rescue_bootcmd; run pogo_bootcmd; reset
bootdelay=3
console=ttyS0,115200
ethact=egiga0
ethaddr=xx:xx:xx:xx:xx:xx
```

```
force_rescue=0
force_rescue_bootcmd=if test $force_rescue -eq 1 || ext2load usb 0:1
0x1700000 /rescueme 1 || fatload usb 0:1 0x1700000 /rescueme.txt 1; then run
rescue_bootcmd; fi
initrd=uinitrd
kernel=uImage
led_error=orange blinking
led_exit=green off
led_init=green blinking
mainlineLinux=yes
mtdids=nand0=orion_nand
mtdparts=mtdparts=orion_nand:1M(u-boot),4M(uImage),32M(rootfs),-(data)
partition=nand0,2
pogo_bootcmd=if fsload uboot-original-mtd0.kwb; then go 0x800200; fi
rescue_bootcmd=if test $rescue_installed -eq 1; then run
rescue_set_bootargs; nand read.e 0x800000 0x100000 0x400000; bootm 0x800000;
else run pogo_bootcmd; fi
rescue_installed=0
rescue_set_bootargs=setenv bootargs console=$console ubi.mtd=2
root=ubi0:rootfs ro rootfstype=ubifs $mtdparts $rescue_custom_params
root_fs=root=LABEL=root
rootfstype=rootfs=ext3
safearcNumber=2097
setbootargs=setenv bootargs console=${console} ${mtdparts} ${root_fs} ro
${rootfstype}
stderr=serial
stdin=serial
stdout=serial
ubifs_bootcmd=run ubifs_set_bootargs; if ubi part data && ubifsmount rootfs
&& ubifsload 0x800000 /boot/uImage && ubifsload 0x1100000 /boot/uInitrd;
then bootm 0x800000 0x1100000; fi
ubifs_mtd=3
ubifs_set_bootargs=setenv bootargs console=$console ubi.mtd=$ubifs_mtd
root=ubi0:rootfs rootfstype=ubifs $mtdparts $ubifs_custom_params
usb_dev_list=3 2 1 0
usb_part_list=4 3 2 1
usb_rootdelay=10
usb_scan=sleep 5; setenv usb_boot_dev none; for dev in $usb_dev_list; do for
part in $usb_part_list; do if ext2load usb ${dev}:$part 0x800000
/boot/$kernel 10; then setenv usb_boot_dev $dev:$part; setenv usb_boot_dir
/boot; fi; if ext2load usb ${dev}:$part 0x800000 /$kernel 10; then setenv
usb_boot_dev $dev:$part; setenv usb_boot_dir ; fi; done; done; if test
"$usb_boot_dev" = "none"; then echo "No USB bootable device found"; else
echo "USB device $usb_boot_dev is bootable"; run setbootargs; echo
$bootargs; sleep 1; ext2load usb $usb_boot_dev 0x800000
$usb_boot_dir/$kernel && setenv usb_boot_address 0x800000; ext2load usb
$usb_boot_dev 0x1100000 $usb_boot_dir/$initrd && setenv usb_boot_address
0x800000 0x1100000; bootm $usb_boot_address; fi;
```

Fontes

- Originally written by louigi600
- Translation PT-BR by MacgyverPT (Miguel Rosa)

howtos, arm, author , louigi600, macgyverpt
translated pt, macgyverpt