Raspberry Pi Network Boot

@Phenomer

October 22, 2014

1 概要

Raspberry Pi でカーネル設定のテスト等を行う際，その都度 SD カードを書き換えるのは面倒である．そこで，ネットワーク経由でカーネルと initrd をダウンロードし起動を行う，ネットワークブート環境を構築する．

2 ディレクトリ構成

構築作業は，基本的に/srv/pxe 以下で行う．(勿論環境に合わせて適宜変更可能である)

- /srv/pxe
- /srv/pxe/tftp - TFTP サーバルート
- /srv/pxe/tftp/pxelinux.cfg - ブート設定ファイル
- /srv/pxe/repo - 各種ソースコード
- /srv/pxe/initrd - initrd 作成作業

```
% sudo mkdir -p /srv/pxe/tftp/pxelinx.cfg /srv/pxe/repo
% sudo chown -R miku:miku /srv/pxe
```

3 ソースコード，ファームウェアの入手

以下のリポジトリを/srv/pxe/repo 以下に clone しておく．

- raspberry pi/firmware
  git://github.com/raspberrypi/firmware.git
- raspberry pi/linux
  git://github.com/raspberrypi/linux.git
- busybox
  git://busybox.net/busybox.git
- u-boot
  git://github.com/swarren/u-boot.git

```
% cd /srv/pxe/repo
% git clone --depth 1                git://github.com/raspberrypi/firmware.git
% git clone --depth 1 -b rpi-3.12.y  git://github.com/raspberrypi/linux.git
% git clone --depth 1 -b 1_22_stable git://git.busybox.net/busybox.git
% git clone --depth 1 -b rpi_dev     git://github.com/swarren/u-boot.git
```

4 SD カード上での設定

Raspberry Pi に接続する SD カードへのファームウェアとブートローダのインストールを行う．

4.1 Partitioning

SD カードに，2048 セクタ目から 32MB 分 FAT16(LBA) でフォーマットされたブート用パーティションを作成する．

```
echo "o\nn\np\n1\n2048\n+32M\nt\ne\nw\n" | sudo fdisk /dev/mmcblk0
sudo mkfs.vfat -F 16 -n RPIBOOT /dev/mmcblk0p1
```

4.2 firmware

/srv/pxe/repo/firmware/boot ディレクトリの中身をブート用パーティションにコピーする．

```
% sudo mount -o uid=miku /dev/mmcblk0p1 /mnt
% cp -r /srv/pxe/firmware/boot/* /mnt
% sudo umount /mnt
```

4.3 u-boot

ネットワークブートに対応した u-boot をビルドする．ビルドには，Raspberry Pi 用のバイナリが出力できるコンパイラが必要である．ここでは，x86_64 上で動作するクロスコンパイラを導入し利用する．[1] Raspberry Pi 上で直接ビルドしても良いが，かなり時間が掛かる．

最初に，make rpi_b_defconfig を実行し，Raspberry Pi 用に設定を初期化する．

```
% cd /srv/pxe/repo/u-boot
% CROSS_COMPILE=/usr/armv6-rpi-linux-gnueabi/bin/armv6-rpi-linux-gnueabi- \
> make rpi_b_defconfig
% CROSS_COMPILE=/usr/armv6-rpi-linux-gnueabi/bin/armv6-rpi-linux-gnueabi- \
> make nconfig
```

nconfig を実行した際に，"Networking support"を有効にし設定を保存する．その後，ビルドを開始する．

```
% CROSS_COMPILE=/usr/armv6-rpi-linux-gnueabi/bin/armv6-rpi-linux-gnueabi- \
> make -j3 u-boot.bin
```

ビルドした u-boot.bin を SD カードのブート領域に kernel.img として保存する．

```
% sudo mount -o uid=miku /dev/mmcblk0p1 /mnt
% cp u-boot.bin /mnt/kernel.img
% sudo umount /mnt
```

この時点で，作成した SD カードを Raspberry Pi に差し込み起動すると，以下のように出力される．

[1] Embedded Linux Wiki - Cross compiling from Linux
http://elinux.org/Raspberry_Pi_Kernel_Compilation#2._Cross_compiling_from_Linux

```
U-Boot 2014.10-rc2-NEGIPI-gf969246 (Oct 11 2014 - 21:19:44)

DRAM:  448 MiB
WARNING: Caches not enabled
MMC:   bcm2835_sdhci: 0
Using default environment

In:    serial
Out:   lcd
Err:   lcd
Net:   Net Initialization Skipped
No ethernet found.
reading /uEnv.txt
 ** Unable to read file /uEnv.txt **
Hit any key to stop autoboot:  0
switch to partitions #0, OK
mmc0 is current device
Scanning mmc 0...
(Re)start USB...
USB0:   Core Release: 2.80a
scanning bus 0 for devices... 3 USB Device(s) found
       scanning usb for storage devices... 0 Storage Device(s) found
       scanning usb for ethernet devices... 1 Ethernet Device(s) found

USB device 0: unknown device
Waiting for Ethernet connection... unable to connect.
missing environment variable: pxeuuid
missing environment variable: bootfile
Retrieving file: pxelinux.cfg/00000000
..
U-Boot>
```

4.4 uEnv.txt の作成

エラーメッセージにあった uEnv.txt を作成し，pxeuuid と bootfile を設定する．

```
% sudo mount -o uid=miku /dev/mmcblk0p1 /mnt
% echo pxeuuid=‘uuidgen‘ > /mnt/uEnv.txt
% echo bootfile=dummy >> /mnt/uEnv.txt
% cat /mnt/uEnv.txt
pxeuuid=b02e1057-df7f-4674-8dd6-b07d7b5a9195
bootfile=dummy
% sudo umount /mnt
```

ここで設定した pxeuuid は，pxelinux.cfg を設定する際に必要となる為，記憶しておく．
uEnv.txt を書き込んだ後に，Raspberry Pi に差し込み起動すると，以下のように出力される．

```
U-Boot 2014.10-rc2-NEGIPI-gf969246 (Oct 11 2014 - 21:19:44)

DRAM:  448 MiB
WARNING: Caches not enabled
MMC:   bcm2835_sdhci: 0
Using default environment

In:    serial
Out:   lcd
Err:   lcd
Net:   Net Initialization Skipped
No ethernet found.
reading /uEnv.txt
60 bytes read in 9 ms (5.9 KiB/s)
Hit any key to stop autoboot:  0
switch to partitions #0, OK
mmc0 is current device
Scanning mmc 0...
(Re)start USB...
USB0:   Core Release: 2.80a
scanning bus 0 for devices... 3 USB Device(s) found
       scanning usb for storage devices... 0 Storage Device(s) found
       scanning usb for ethernet devices... 1 Ethernet Device(s) found

USB device 0: unknown device
Waiting for Ethernet connection... unable to connect.
Retrieving file: pxelinux.cfg/b02e1057-df7f-4674-8dd6-b07d7b5a9195
..
U-Boot>
```

SD カード上で必要な設定は以上である．

5　カーネルのビルド

Raspberry Pi 用の busybox をビルドする．u-boot と同様に，クロスコンパイラを用いて x86_64 の PC 上で作業を行う．

"bcmrpi_cutdown_defconfig"を元に，initrd に対応したカーネルを作成する．

```
% cd /srv/pxe/repo/linux
% ARCH=arm \
> CROSS_COMPILE=/usr/armv6-rpi-linux-gnueabi/bin/armv6-rpi-linux-gnueabi- \
> make bcmrpi_cutdown_defconfig
% ARCH=arm \
> CROSS_COMPILE=/usr/armv6-rpi-linux-gnueabi/bin/armv6-rpi-linux-gnueabi- \
> make localyesconfig
% ARCH=arm \
> CROSS_COMPILE=/usr/armv6-rpi-linux-gnueabi/bin/armv6-rpi-linux-gnueabi- \
> make nconfig
```

nconfig を実行したら，"General setup/Initial RAM filesystem and RAM disk (initramfs/initrd) support"を有効にし保存する．

```
103c103,110
< # CONFIG_BLK_DEV_INITRD is not set
---
> CONFIG_BLK_DEV_INITRD=y
> CONFIG_INITRAMFS_SOURCE=""
> CONFIG_RD_GZIP=y
> # CONFIG_RD_BZIP2 is not set
> # CONFIG_RD_LZMA is not set
> # CONFIG_RD_XZ is not set
> # CONFIG_RD_LZO is not set
> # CONFIG_RD_LZ4 is not set
2091a2099
> CONFIG_DECOMPRESS_GZIP=y
```

完成した arch/arm/boot/zImage を，/srv/pxe/tftp 以下にコピーしておく．

```
% ARCH=arm \
> CROSS_COMPILE=/usr/armv6-rpi-linux-gnueabi/bin/armv6-rpi-linux-gnueabi- \
> make -j3 zImage
% cp arch/arm/boot/zImage /srv/pxe/tftp
```

6 initrd の作成

busybox のみを使った最小限の rootfs を作成し，起動後にすぐ sh を叩けるようにする．

6.1 busybox のビルド

Raspberry Pi 用の busybox をビルドする．u-boot やカーネルと同様に，クロスコンパイラを用いて x86_64 の PC 上で作業を行う．

まず，make defconfig で設定を初期化する．

```
% cd /srv/pxe/repo/busybox
% CROSS_COMPILE=/usr/armv6-rpi-linux-gnueabi/bin/armv6-rpi-linux-gnueabi- \
> make defconfig
```

次に，Busybox Settings/Build Options/Build BusyBox as a static binary (no shared libs) を有効にし，設定を保存後，ビルドする．

```
% CROSS_COMPILE=/usr/armv6-rpi-linux-gnueabi/bin/armv6-rpi-linux-gnueabi- \
> make menuconfig
% CROSS_COMPILE=/usr/armv6-rpi-linux-gnueabi/bin/armv6-rpi-linux-gnueabi- \
> make -j3 busybox
```

ビルドした Raspberry Pi 用 busybox を/srv/pxe/busybox.armv6l にコピーする．

```
% file busybox
busybox: ELF 32-bit LSB executable, ARM, EABI5 version 1 (SYSV), statically linked,
for GNU/Linux 3.2.27, BuildID[sha1]=0132e3df4d17dda946f187ee92a531c319403694, stripped
% cp busybox /srv/pxe/busybox.armv6l
```

6.2 initrd ビルドスクリプトの作成

initrd を作成するための作業をシェルスクリプトにしたものは，以下の通りである．

```
#!/bin/sh

WORKDIR="/srv/pxe"
ROOTFS="/srv/pxe/initrd"
INITRD="/srv/pxe/tftp/initrd.gz"
BBOX="/srv/pxe/busybox.armv6l"

rm -rf ${ROOTFS}
mkdir -p ${ROOTFS}/bin ${ROOTFS}/dev ${ROOTFS}/proc \
         ${ROOTFS}/sys ${ROOTFS}/tmp ${ROOTFS}/root
cp ${BBOX} ${ROOTFS}/bin/busybox

cd ${ROOTFS}/bin
for CMD in `busybox --list`; do
    ln -s busybox ${CMD}
done
cd ${WORKDIR}

cat <<EOF > ${ROOTFS}/init
#!/bin/sh

mount -t devtmpfs  devtmpfs  /dev
mount -t proc      proc      /proc
mount -t sysfs     sysfs     /sys
mount -t tmpfs     tmpfs     /tmp -o mode=1777
setsid cttyhack /bin/sh
EOF
chmod +x ${ROOTFS}/init

cd ${ROOTFS}
find . | cpio -o -H newc | gzip -c > ${INITRD}
cd ${WORKDIR}

echo done.
```

これを実行すると，/srv/pxe/tftp/initrd.gz が出力される．

```
% ./initrd.sh
3607 blocks
done.
% file tftp/initrd.gz
tftp/initrd.gz: gzip compressed data,
 last modified: Sat Oct 11 23:03:24 2014, from Unix
```

7　サーバ側の構築

サーバ側には，ネットワーク設定を割り当てる DHCP サーバとカーネル等のファイルを配布する TFTP サーバが必要になる為，これらを構築する．

7.1　DHCP

DHCP サーバには，busybox udhcpd を利用する．

7.1.1　/srv/pxe/udhcpd.conf

udhcpd.conf は以下の通りである．siaddr が TFTP サーバのアドレス，boot_file が最初に読み込まれる起動用ファイルである．今回は，pxelinux.cfg 以下のファイルを用いてカーネルと initrd を読み込むように構築する為，0byte のダミーファイルを用意しておく．

```
interface enp2s0

start 10.3.9.10
end   10.3.9.20

siaddr    10.3.9.1
boot_file dummy

opt     dns     10.3.9.1
opt     router  10.3.9.1
option  subnet  255.255.255.0
option  domain  k.hachune.org
option  lease   864000
```

7.1.2　udhcpd の起動

```
% touch /srv/pxe/tftp/dummy
% sudo busybox udhcpd -f /srv/pxe/udhcpd.conf
```

7.2　TFTP

TFTP サーバには，tftp-hpa を用いる．/srv/pxe/tftp を TFTP サーバのルートとして利用するよう設定し実行する．

```
% sudo in.tftpd -l -L -s -v /srv/pxe/tftp
```

7.3　pxelinux.cfg

ブート設定ファイルは，/srv/pxe/tftp/pxelinux.cfg 以下に記述する．ファイル名は，SD カードの uEnv.txt に記述した UUID を用いる．(/srv/pxe/tftp/pxelinux.cfg/b02e1057-df7f-4674-8dd6-b07d7b5a9195)

```
default rpibbox

label rpibbox
  menu label linux(busybox)
  kernel /zImage
  append earlyprintk console=ttyAMA0,115200 rootwait
  initrd /initrd.gz
```

8 起動

ここまでの構築・設定により，Raspberry Pi をネットワークに接続し起動すると，以下のように自動的にネットワーク設定を行い，ネットワーク経由でカーネルと initrd，カーネル起動パラメータを読み込み，起動できるようになる．

```
U-Boot 2014.10-rc2-NEGIPI-gf969246 (Oct 11 2014 - 21:19:44)

DRAM:  448 MiB
WARNING: Caches not enabled
MMC:   bcm2835_sdhci: 0
Using default environment

In:    serial
Out:   lcd
Err:   lcd
Net:   Net Initialization Skipped
No ethernet found.
reading /uEnv.txt
60 bytes read in 9 ms (5.9 KiB/s)
Hit any key to stop autoboot:  0
switch to partitions #0, OK
mmc0 is current device
Scanning mmc 0...
(Re)start USB...
USB0:   Core Release: 2.80a
scanning bus 0 for devices... 3 USB Device(s) found
       scanning usb for storage devices... 0 Storage Device(s) found
       scanning usb for ethernet devices... 1 Ethernet Device(s) found

USB device 0: unknown device
Waiting for Ethernet connection... done.
BOOTP broadcast 1
DHCP client bound to address 10.3.9.20 (69 ms)
Using sms0 device
TFTP from server 10.3.9.1; our IP address is 10.3.9.20
Filename 'dummy'.
Load address: 0x200000
Loading: #
         0 Bytes/s
done
Retrieving file: pxelinux.cfg/b02e1057-df7f-4674-8dd6-b07d7b5a9195
Waiting for Ethernet connection... done.
Using sms0 device
TFTP from server 10.3.9.1; our IP address is 10.3.9.20
Filename 'pxelinux.cfg/b02e1057-df7f-4674-8dd6-b07d7b5a9195'.
Load address: 0x100000
Loading: #
         36.1 KiB/s
done
Bytes transferred = 150 (96 hex)
Config file found
1:      linux(busybox)
```

```
Retrieving file: /initrd.gz
Waiting for Ethernet connection... done.
Using sms0 device
TFTP from server 10.3.9.1; our IP address is 10.3.9.20
Filename '/initrd.gz'.
Load address: 0x2100000
Loading: ##################################################################
         #####
         1.8 MiB/s
done
Bytes transferred = 1019593 (f8ec9 hex)
Retrieving file: /zImage
Waiting for Ethernet connection... done.
Using sms0 device
TFTP from server 10.3.9.1; our IP address is 10.3.9.20
Filename '/zImage'.
Load address: 0x1000000
Loading: ##################################################################
         ##################################################################
         ##################################################################
         ############################
         1.8 MiB/s
done
Bytes transferred = 3253872 (31a670 hex)
append: earlyprintk console=ttyAMA0,115200 rootwait
Kernel image @ 0x1000000 [ 0x000000 - 0x31a670 ]

Starting kernel ...

Uncompressing Linux... done, booting the kernel.
Booting Linux on physical CPU 0x0
Linux version 3.12.29+ (miku@hachune.k.hachune.org)
(gcc version 4.8.2 20130603 (prerelease) (crosstool-NG 1.19.0) ) #3 Wed Oct 8 20:36:24 JST 2014
...
/ # uname -a
Linux (none) 3.12.29+ #3 Wed Oct 8 20:36:24 JST 2014 armv6l GNU/Linux
```

9 おわりに

ネットワーク経由でカーネルと initrd，カーネル起動パラメータを読み込み起動できるようになった．これにより，カーネル設定のテスト等といった作業上の問題以外にも，NFS root や iSCSI root 等，Raspberry Pi の弱点である SD カードの寿命に関する問題を回避する手段を用いる事も可能になった．

10 参考

- Embedded Linux Wiki - RPi U-Boot
  `http://elinux.org/RPi_U-Boot`
- lentinj/u-boot - u-boot/doc/README.pxe
  `https://github.com/swarren/u-boot/blob/rpi_dev/doc/README.pxe`
- busybox - root/examples/udhcp/udhcpd.conf
  `http://git.busybox.net/busybox/tree/examples/udhcp/udhcpd.conf`