Kernel Masters

http://www.kernelmasters.org

Creating a Root File System for Linux on OMAP4460

How to Create a Root File System

The root file system built in this note is based on BusyBox.

The root filesystem may be kept in RAM (loaded from a (usually compressed) image in ROM or flash), or on a disk-based filesystem (stored in ROM or flash), or loaded from the network (often over TFTP) if applicable. If the root filesystem is in RAM, make it the initramfs — a RAM filesystem whose content is created at boot time.

Many frameworks exist for assembling root images for embedded systems. There are a few pointers in the BusyBox FAQ. Buildroot is a popular one, allowing you to build a whole root image with a setup similar to the Linux kernel and BusyBox. OpenEmbedded is another such framework.

http://en.wikipedia.org/wiki/Linux_startup_process

http://en.wikipedia.org/wiki/Runlevel

What is Busy Box?

BusyBox combines tiny versions of many common UNIX utilities into a single small executable. It provides replacements for most of the utilities you usually find in GNU fileutils, shellutils, etc. The utilities in BusyBox generally have fewer options than their full-featured GNU cousins; however, the options that are included provide the expected functionality and behave very much like their GNU counterparts. BusyBox provides a fairly complete environment for any small or embedded system.

BusyBox has been written with size-optimization and limited resources in mind. It is also extremely modular so you can easily include or exclude commands (or features) at compile time. This makes it easy to customize your embedded systems. To create a working system, just add some device nodes in /dev, a few configuration files in /etc, and a Linux kernel.

Busy Box Setup

Step 1: Download Busy Box Source code from local server

$ cd ~/KM_GIT/

$ git clone git@192.168.1.3:busybox.git

$ cd busybox

Step 2: Configuration

$ make menuconfig

Busybox Settings-> General Configuration-> Show applet usage messages

Busybox Settings-> General Configuration-> Store applet usage messages in compressed form

Busybox Settings-> General Configuration-> Use the devpts filesystem for Unix98 PTYs

Busybox Settings-> General Configuration-> (/proc/self/exe) Path to BusyBox executable

Busybox Settings-> Build Options-> Build BusyBox as a static binary (no shared libs)

Busybox Settings-> Build Options-> Build with Large File Support

Busybox Settings-> Build Options-> Do you want to build BusyBox with a Cross Compiler

(/home/kernelmasters/toolchain/gcc-linaro-arm-linux-gnueabihf-4.7-2013.02-01-20130221_linux/bin/arm-linux-gnueabihf-) Cross Compiler prefix

Disable RPC server:

Menuconfig-> Networking Utilities -> Support RPC services.

Step 3: Compilation

$ make (after few minutes busybox binary file generated)

Step 4: Installation

```
$ sudo su

root # mkdir ~/KM_GIT/rootfs

root # make CONFIG_PREFIX=~/KM_GIT/rootfs install
```

Creating a Root File System

```
root # cd ~/KM_GIT/rootfs/

root # mkdir dev

root # mknod dev/console c 5 1

root # mkdir dev/pts

root # mkdir etc

root # mkdir etc/init.d

root # mkdir lib

root # mkdir mnt

root # mkdir opt

root # mkdir proc

root # mkdir root

root # mkdir sys

root # mkdir tmp

root # mkdir var

root # mkdir var/log

root # mkdir debug

root # cd etc

root # vi fstab

proc /proc proc defaults 0 0 none /dev/pts devpts mode=0622 0 0

root # vi group

root:x:0:root
```

```
root # vi passwd
```

```
root::0:0:root:/root:/bin/ash
```

```
root # vi hosts
```

```
127.0.0.1 localhost
```

The kernel starts “/sbin/init” after it boots (actually the kernel attempts to execute several known programs until one succeeds). Init reads the etc/inittab file to determine what to do at start up, shutdown, or when a user logs in. These inittab files can get quite complicated. A simple one is shown below:

```
root # vi inittab
```

- inittab: inittab

The “sysinit” line tells init to run the /etc/init.d/rcS script to set up the system. The rest are self explanatory.

```
root # vi init.d/rcS
```

- rcS: rcS

```
root # chmod +x init.d/rcS
```

http://www.kernelmasters.org

-- AdminUser - 04 Oct 2014

- initrd.txt: initrd.txt

Edit | Attach | Print version | History: r2 < r1 | Backlinks | View wiki text | Edit wiki text | More topic actions

Topic revision: r2 - 04 Oct 2014, AdminUser