

Building Linux for the OMAP™ Software Development Platform

Steve Kipisz

ABSTRACT

This report describes the process of building and running Linux on the OMAP Software Development Platform. Tool chain and root file system installation are also included for kernel testing purposes.

Contents

1	Introduction	1
2	Requirements	1
	2.1 Hardware Requirements	1
	2.2 Software Requirements	2
3	OMAPxxxx Software Development Platform Boot loader	2
4	Getting the Tool Chain, Kernel Source, and Root File System	2
5	Installing the Tool Chain.....	2
6	Installing the Kernel Source.....	3
7	Installing the Root File System.....	3
8	Build the Linux Kernel Image.....	3
9	Format the Kernel Image for U-Boot Download	4
10	Download and Boot the Kernel	4
	References.....	6

1 Introduction

Linux for the OMAP Software Development Platform can be built either by obtaining an ARM tool chain and kernel source from Monta Vista, or by installing an Open Source tool chain and building an Open Source Linux kernel. This report describes the Open Source process.

2 Requirements

2.1 Hardware Requirements

- OMAPxxxx Software Development Platform (SDP)

Overwrite this text with the Lit. Number

- Serial null-modem cable or serial cable with a null-modem adapter
- PC running RedHat™ Linux

RedHat™ Linux 9.0 was installed on the PC used to build the kernel for this report.

2.2 Software Requirements

- Open Source arm linux tool chain
- Open Source Linux kernel source with OMAP support
- ARM Linux root file system

The OMAP1610 Software Development Platform (also know as “H2 Sample”) was used to test the kernel built.

3 OMAPxxxx Software Development Platform Boot loader

This report assumes the u-boot boot loader is already installed. Building and installing u-boot is out of the scope of this report and is described in “*U-Boot for the OMAP16xx GSM/GPRS Software Development Platform*” [1].

4 Getting the Tool Chain, Kernel Source, and Root File System

Open up your web browser and go to <http://linux.omap.com>. Select the H2 Board and download the tool chain, kernel source, and root file system for the 2.6 Sample Build. The file names are:

Tool Chain	gcc-3.4.0.tar.bz2
Kernel Source	2.6_kernel_041404.tar.bz2
Root File System	rootfs.tar.bz2

5 Installing the Tool Chain

This report follows the same directory conventions as “*Building Linux for the Innovator Development Kit for OMAP™ Platform*” [2]. The source files are downloaded to the /root directory while the tool chain will be installed into the /opt directory.

Change directory to the /opt directory and un-tar the tool chain file.

```
[root@localhost]# cd /opt
[root@localhost]# tar -xjvf /root/gcc-3.4.0.tar.bz2
```

The tool chain will be installed in the /opt/gcc-3.4.0 directory.

6 Installing the Kernel Source

Change directory to the /usr/src directory and un-tar the kernel source file.

```
[root@localhost]# cd /usr/src
[root@localhost]# tar -xjvf /root/2.6_kernel_041404.tar.bz2
```

The kernel source tree will be installed in the /usr/src/2.6_kernel directory.

7 Installing the Root File System

The root file system is hosted on the PC and will be NFS mounted by the kernel at boot up. Create a /data directory, change directory to it, and then un-tar the root file system file.

```
[root@localhost]# mkdir /data
[root@localhost]# cd /data
[root@localhost]# tar -xjvf /root/rootfs.tar.bz2
```

The root file system will be installed in the /data/rootfs directory.

Setting up the PC as an NFS server is out of the scope of this document. See the Linux man pages or any book on installing and configuring Red Hat Linux.

8 Build the Linux Kernel Image

Make sure the path to the tool chain is in the current path. The kernel is built in two steps. The first configures the kernel to build for the H2 Sample while the second builds the kernel.

```
[root@localhost]# cd /usr/src/2.6_kernel
[root@localhost]# export PATH=/opt/gcc-3.4.0/bin:$PATH
[root@localhost]# make omap_1610_h2_defconfig
[root@localhost]# make
```

The **export** command adds the tool chain path to the current path. The next command configures to build the H2 Sample kernel, while the final **make** makes the kernel.

Overwrite this text with the Lit. Number

9 Format the Kernel Image for U-Boot Download

The file `arch/arm/boot/compressed/vmlinux` is the kernel image, but it is not in a suitable format for U-Boot to download. To convert the file, the `mkimage` tool that comes with U-Boot must be used. See [1] for more information on building and installing U-Boot.

```
[root@localhost]# cd /usr/src/2.6_kernel
[root@localhost]# arm-linux-objcopy -O binary -R .note -R .comment -S
arch/arm/boot/compressed/vmlinux linux.bin
[root@localhost]# gzip -9 linux.bin
[root@localhost]# mkimage -A arm -O linux -T kernel -C gzip -a 0x10c08000 -e 0x10c08000
-n "Linux Kernel Image" -d linux.bin.gz uImage.cc
```

10 Download and Boot the Kernel

Downloading and booting a kernel is described in [1]. Refer to that document on setting up the SDP and PC for downloading and booting this kernel. After booting the kernel, the terminal emulator should show something like below

```

U-Boot 1.0.0 (Mar  9 2004 - 14:18:33)

U-Boot code: 11000000 -> 110143AC  BSS: -> 110180AC
DRAM Configuration:
Bank #0: 10000000 32 MB
Flash: 32 MB
In:    serial
Out:   serial
Err:   serial
OMAP1610 H2 # bootm 100000
## Booting image at 00100000 ...
   Image Name:   Linux Kernel Image
   Image Type:   ARM Linux Kernel Image (gzip compressed)
   Data Size:    844543 Bytes = 824.7 kB
   Load Address: 10c08000
   Entry Point:  10c08000
   Verifying Checksum ... OK
   Uncompressing Kernel Image ... OK

Starting kernel ...

Uncompressing Linux..... done,
booting the kernel.
Linux version 2.6.4-omap1 (root@dta0867407.sc.ti.com) (gcc version 3.4.0 20031230
(CodeSourcery ARM Q4 2003)) #1 Thu May 13 13:22:25 CDT 2004
CPU: ARM926EJ-Sid(wb) [41069263] revision 3 (ARMv5TEJ)
CPU: D VIPT write-back cache
CPU: I cache: 16384 bytes, associativity 4, 32 byte lines, 128 sets
CPU: D cache: 8192 bytes, associativity 4, 32 byte lines, 64 sets
Machine: TI-Innovator/OMAP1610
Memory policy: ECC disabled, Data cache writeback
On node 0 totalpages: 8192
  DMA zone: 8192 pages, LIFO batch:2
  Normal zone: 0 pages, LIFO batch:1
  HighMem zone: 0 pages, LIFO batch:1
Built 1 zonelists
Kernel command line: mem=32M noinitrd console=ttyS0,115200 ip=dhcp root=/dev/nfs rw
nfsroot=128.247.75.227:/data/rootfs,nolock
PID hash table entries: 256 (order 8: 2048 bytes)
Console: colour dummy device 80x30
Memory: 32MB = 32MB total
Memory: 30512KB available (1417K code, 320K data, 92K init)
Calibrating delay loop... 83.76 BogoMIPS
Dentry cache hash table entries: 4096 (order: 2, 16384 bytes)
Inode-cache hash table entries: 2048 (order: 1, 8192 bytes)
Mount-cache hash table entries: 512 (order: 0, 4096 bytes)
CPU: Testing write buffer coherency: ok
POSIX conformance testing by UNIFIX
NET: Registered protocol family 16
omap1610 FB 2004 by Dirk Behme
OMAP DMA hardware version 1
DMA capabilities: 000c0000:00000000:01ff:003f:007f
OMAP virtual buses initialized
OMAP GPIO hardware version 1.0
OMAP OCPI interconnect driver loaded
NetWinder Floating Point Emulator V0.97 (double precision)
devfs: 2004-01-31 Richard Gooch (rgooch@atnf.csiro.au)
devfs: boot_options: 0x0
Console: switching to colour frame buffer device 30x40
omap-rtc: RTC power up reset detected.
omap-rtc: Enabling RTC.

```

Overwrite this text with the Lit. Number

```

Real Time Clock Driver v1.0
omap_wdt: TI OMAP Watchdog Timer: timer margin 32 sec
Serial: 8250/16550 driver $Revision: 1.90 $ 7 ports, IRQ sharing disabled
ttyS0 at MMIO 0xffffb0000 (irq = 46) is a OMAP UART
ttyS1 at MMIO 0xffffb0800 (irq = 47) is a OMAP UART
ttyS2 at MMIO 0xffffb9800 (irq = 15) is a OMAP UART
RAMDISK driver initialized: 16 RAM disks of 4096K size 1024 blocksize
smc9194.c:v0.15 12/15/00 by Erik Stahlman (erik@vt.edu)
eth0: SMC91C94/91C96 (rev 9) at 0xe8000000 IRQ:96 INTF:TP MEM:6144b ADDR:
00:50:c2:27:15:5f
PPP generic driver version 2.4.2
SLIP: version 0.8.4-NET3.019-NEWTTY (dynamic channels, max=256).
CSLIP: code copyright 1989 Regents of the University of California.
Console: switching to colour frame buffer device 30x40
Registering OMAP device 'omap-lcd1'. Parent at tipb
mice: PS/2 mouse device common for all mice
OMAP Keypad Driver
NET: Registered protocol family 2
IP: routing cache hash table of 512 buckets, 4Kbytes
TCP: Hash tables configured (established 2048 bind 4096)
NET: Registered protocol family 1
NET: Registered protocol family 17
Sending DHCP requests ., OK
IP-Config: Got DHCP answer from 255.255.255.255, my address is 128.247.75.161
IP-Config: Complete:
    device=eth0, addr=128.247.75.161, mask=255.255.254.0, gw=128.247.74.1,
    host=128.247.75.161, domain=sc.ti.com, nis-domain=(none),
    bootserver=255.255.255.255, rootserver=128.247.75.227, rootpath=
Looking up port of RPC 100003/2 on 128.247.75.227
Looking up port of RPC 100005/1 on 128.247.75.227
VFS: Mounted root (nfs filesystem).
Freeing init memory: 92K
init started: BusyBox v1.00-pre8 (2004.03.05-22:18+0000) multi-call binary

*****
Starting System Init for innovator
*****

Please press Enter to activate this console.

BusyBox v1.00-pre8 (2004.03.05-22:18+0000) Built-in shell (ash)
Enter 'help' for a list of built-in commands.

-sh: can't access tty; job control turned off
#

```

References

1. *U-Boot for the OMAP16xx GSM/GPRS Software Development Platform*
2. *Building Linux for the Innovator Development Kit for OMAP™ Platform (SWPA011)*