

U-Boot for the OMAP16xx GSM/GPRS Software Development Platform



ABSTRACT

This report describes the process of building and running **U-Boot** on the OMAP16xx GSM/GPRS Software Development Platform (SDP). U-Boot is an open source bootloader for embedded development platforms, which can be installed in a boot ROM/flash and used to initialize and test the hardware or to download and run application code.

Contents

| | | |
|---|--|-----------|
| 1 | Introduction | 2 |
| 2 | Copyright | 2 |
| 3 | Disclaimer | 2 |
| 4 | Requirements | 3 |
| 4.1 | Hardware Requirements | 3 |
| 4.2 | Software Requirements | 3 |
| 5 | OMAP16xx SDP Switch Configuration | 3 |
| 6 | Getting the U-Boot bootloader | 3 |
| 7 | Building U-Boot | 4 |
| 8 | Host side setup | 4 |
| 8.1 | Windows Based Host | 4 |
| 8.1.1 | Serial Console Access | 4 |
| 8.2 | Linux RedHat Host | 5 |
| 8.2.1 | Serial Console Access | 5 |
| 8.2.2 | Configuration of a TFTP Server | 7 |
| 8.2.3 | Configuration of a BOOTP / DHCP Server | 8 |
| 9 | Flashing U-Boot | 9 |
| 10 | Executing U-Boot | 12 |
| 11 | Building a Linux Kernel Image to download from U-Boot | 14 |
| 12 | Downloading a Kernel Image with U-Boot | 14 |
| 12.1 | Serial Download | 14 |
| 12.2 | TFTP Download | 15 |
| 12.3 | Verifying the Download | 16 |
| 13 | Booting Linux kernel from U-Boot | 17 |
| 14 | Upgrading U-Boot | 17 |
| 15 | References | 18 |
| Appendix A | | 19 |
| Download Linux kernel using tftpboot on Corporate Network | | 19 |

OMAP is a trademark of Texas Instruments Incorporated.
All other trademarks are the property of their respective owners.

1 Introduction

The U-Boot bootloader is designed for embedded development platforms. It can be installed in a boot ROM and is used to initialize and test the hardware. It can also be used to download and run the Linux kernel or any specific application. This bootloader is licensed under GPL and is maintained by Wolfgang Denk [wd@denk.de].

2 Copyright

Copyright (c) 2001 - 2003 by Wolfgang Denk, DENX Software Engineering.

Copyright (c) 2001 – 2004, Texas Instruments Inc.

You have the freedom to distribute copies of this document in any format or to create a derivative work of it and distribute it provided that you:

- Distribute this document or the derivative work at no charge at all. It is **not** permitted to sell this document or the derivative work or to include it into any package or distribution that is not freely available to everybody.
- Send your derivative work (in the most suitable format such as sgml) to the author.
- License the derivative work with this same license or use GPL. Include a copyright notice and at least a pointer to the license used.
- Give due credit to previous authors and major contributors.

It is requested that corrections and/or comments be forwarded to the author.

If you are considering to create a derived work other than a translation, it is requested that you discuss your plans with the author.

3 Disclaimer

Use the information in this document at your own risk. TI disavows any potential liability for the contents of this document. Use of the concepts, examples, and/or other content of this document is entirely at your own risk. All copyrights are owned by their owners, unless specifically noted otherwise. Use of a term in this document should not be regarded as affecting the validity of any trademark or service mark. Naming of particular products or brands should not be seen as endorsements.

4 Requirements

4.1 Hardware Requirements

- OMAP16xx GSM/GPRS Software Development Platform (SDP)
- Serial null-modem cable or serial cable with a null-modem adapter
- PC running Windows 2000/NT
- PC running RedHat™ Linux.

4.2 Software Requirements

- Open source arm linux gcc 3.4.0 compiler tool chain
- IBoot Host and IBoot Console software for the OMAP16xx GSM/GPRS SDP.

5 OMAP16xx SDP Switch Configuration

There are a set of 10 dip switches on the processor module. The switch number 9, labeled as ARM BOOT is used to switch the address mapping of chip selects CS0 and CS3.

| ARM BOOT switch | Address of CS0 (BOOT ROM) | Address of CS3 (USER FLASH) |
|-----------------|---------------------------|-----------------------------|
| ON | 0x0000 0000 | 0x0C00 0000 |
| OFF | 0x0C00 0000 | 0x0000 0000 |

CS0 selects the Boot ROM, while CS3 selects the user Flash.

To boot the device using U-boot, U-boot must be put in flash (CS3) at address 0x00000000 and ARM BOOT switched OFF before powering ON the device.

6 Getting the U-Boot bootloader

TI has added support for the OMAP16xx SDP in this bootloader. The U-Boot bootloader can be obtained from the main tree of U-Boot source.

The U-Boot project is hosted at Sourceforge, where you can find the project home page: <http://sourceforge.net/projects/u-boot/>.

Official releases of U-Boot are also available through FTP. Compressed tar archives can be downloaded from the directory <ftp://ftp.denx.de/pub/u-boot/>.

7 Building U-Boot

Building U-Boot requires a Linux Host machine. The ARM gcc cross compiler version 3.4.0 is required for the compilation and linking of U-Boot. The pre-built cross compiler tools can be downloaded from <http://linux.omap.com/pub/toolchain/gcc-3.4.0.tar.bz2>. An installation detail on the arm-linux tool chain is beyond the scope of this document. Please refer to *Building Linux for the OMAP™ Software Development Platform [2]* for more information.

To build U-Boot, use the following commands

```
[root@localhost]# cd u-boot-1.0.2
[root@localhost]# make distclean
[root@localhost]# make omap1610h2_config
[root@localhost]# make
```

This should create the following files as output

u-boot – in **ELF** format

u-boot.bin – a raw binary image

u-boot.srec – in Motorola™ srec format

NOTE: If you plan to boot through the secure ROM, you must build U-boot using

```
[root@localhost]# make omap1610h2_cs0boot_config
[root@localhost]# make
```

8 Host side setup

A host machine is used to communicate with the Target. This machine can be either a Windows based machine or a Linux RedHat™ machine. The setup for both the Operating systems is given below.

8.1 Windows Based Host

8.1.1 Serial Console Access

To use U-Boot with Windows as a development system and to make full use of all capabilities requires access to a serial console port on the target system. There are several ways to access the serial console port, such as using a terminal server, but the most common way is to attach a serial port to the host. Additionally, a "terminal emulation program" is required.

8.1.1.1 Configuring the Terminal Emulation Program

This section assumes that Hyperterminal is used as the terminal emulation program. This can be used to connect to the serial port. Set the com1 settings as

Bits per second – 115200 ; Data bits – 8 ; Parity – None ; Stop bits – 1; Flow Control - None

8.2 Linux RedHat Host

8.2.1 Serial Console Access

To use U-Boot and Linux as a development system and to make full use of all their capabilities you will need access to a serial console port on your target system. There are several ways to access the serial console port, such as using a terminal server, but the most commonly way is to attach a serial port on your host. Additionally you will need some "terminal emulation program".

8.2.1.1 Configuring the "cu" command

The `cu` command is part of the UUCP package and can be used to act as a dial in terminal. It can also do simple file transfers, which can be used in U-Boot for image download.

On RedHat systems you can check if the UUCP package is installed as follows:

```
bash$ rpm -q uucp
uucp-1.06.1-33.7.2
```

If necessary, install the UUCP package from your distribution media.

To configure `cu` for use with U-Boot and Linux please make sure that the following entries are present in the **UUCP** configuration files; depending on your target configuration the serial port and/or the console baud rate may be different from the values used in this example: (`/dev/ttyS0`, 115200 bps, 8N1):

`/etc/uucp/sys`

```
#
# /dev/ttyS0 at 115200 bps:
#
system      S0@115200
port        serial0_115200
time        any
```

`/etc/uucp/port`

```
#
# /dev/ttyS0 at 115200 bps:
#
port        serial0_115200
```

```
type      direct
device    /dev/ttyS0
speed     115200
hardflow  false
```

You can then connect to the serial line using the command

```
bash$ cu S0@115200
Connected.
```

To disconnect, type the escape character '~' followed by '.' at the beginning of a line.

See also: **cu(1)**, **info uucp**.

8.2.1.2 Configuring the "kermit" command

The name **kermit** stands for a whole family of communications software for serial and network connections. The fact that it is available for most computers and operating systems makes it especially well suited for our purposes.

kermit executes the commands in its initialization file, `.kermrc`, in your home directory before it executes any other commands, so this can be easily used to customize its behavior using appropriate initialization commands. The following settings are recommended for use with U-Boot and Linux:

```
~/kermrc
```

```
set line /dev/ttyS0
set speed 115200
set carrier-watch off
set handshake none
set flow-control none
robust
set file type bin
set file name lit
set rec pack 1000
set send pack 1000
```

```
set window 5
```

This example assumes that the first serial port on the host system (`/dev/ttyS0`) is set at a baud rate of 115200 to connect to the target's serial console port.

You can then connect to the serial line:

```
bash$ kermit -c
Connecting to /dev/ttyS0, speed 115200.
The escape character is Ctrl-\ (ASCII 28, FS)
Type the escape character followed by C to get back,
or followed by ? to see other options.
-----
```

Tip: If you cannot find **kermit** on the distribution media for the Linux host system, you can download it from the kermit project page: <http://www.columbia.edu/kermit/>

8.2.2 Configuration of a TFTP Server

The fastest way to use U-Boot to load a Linux kernel or an application image is file transfer over ethernet. For this purpose, U-Boot implements the **TFTP** protocol (see the **tftpboot** command in U-Boot). To enable TFTP support on your host system, you must make sure that the TFTP daemon program `/usr/sbin/in.tftpd` is installed. On Red Hat Linux release 7.3 (Valhalla) systems you can verify this as follows:

```
bash$ rpm -q tftp-server
tftp-server-0.28-2
```

If necessary, install the TFTP daemon program from your distribution media.

Most Linux distributions disable the TFTP service by default. To enable it for example on Red Hat Linux release 7.3 (Valhalla) systems, edit the file `/etc/xinetd.d/tftp` and remove the line

```
Disable = yes
```

or change it into a comment line by putting a hash character in front of it:

```
bash$ cat /etc/xinetd.d/tftp
# default: off
# description: The tftp server serves files using the trivial file transfer \
#           protocol. The tftp protocol is often used to boot diskless \
```

```
#      workstations, download configuration files to network-aware printers, \
#      and to start the installation process for some operating systems.

service tftp
{
    socket_type      = dgram
    protocol         = udp
    wait             = yes
    user             = root
    server           = /usr/sbin/in.tftpd
    server_args      = -s /tftpboot
    disable          = yes
    per_source       = 11
    cps              = 100 2
}

```

Also, make sure that the /tftpboot directory exists and is world-readable (permissions at least "dr-xr-x-r-x").

8.2.3 Configuration of a BOOTP / DHCP Server

BOOTP / DHCP can be used to automatically pass configuration information to the target. The only information the target must "know" about itself is its ethernet (MAC) address. The following command can be used to check if DHCP support is available on the host system:

```
bash$ rpm -q dhcp
dhcp-2.0pl5-8
```

If necessary, install the DHCP package from your distribution media and create the DHCP configuration file /etc/dhcpd.conf that matches the network setup. The following provides an example:

```
subnet 192.168.1.0 netmask 255.255.255.0 {
    option routers      192.168.1.1;
    option subnet-mask  255.255.255.0;

    option domain-name "local.net";
}
```

```
option domain-name-servers ns.local.net;

host trgt { hardware ethernet 00:30:BF:01:02:D0;
            fixed-address 192.168.1.99;
            option root-path "/opt/eldk/ppc_8xx";
            option host-name "trgt";
            next-server 192.168.1.2;
            filename "/tftpboot/plimage";
        }
}
```

With this configuration, the DHCP server will reply to a request from the target with the ethernet address **00:30:BF:01:02:D0** with the following information:

The target is located in the subnet **192.168.1.0**, which uses the netmask **255.255.255.0**.

The target has the hostname "**trgt**" and the IP address **192.168.1.99**.

The host with the IP address **192.168.1.2** will provide the boot image for the target and provide NFS server function in cases when the target mounts its root filesystem over NFS.

Tip: Note: The host listed with the **next-server** option can be different from the host that is running the DHCP server.

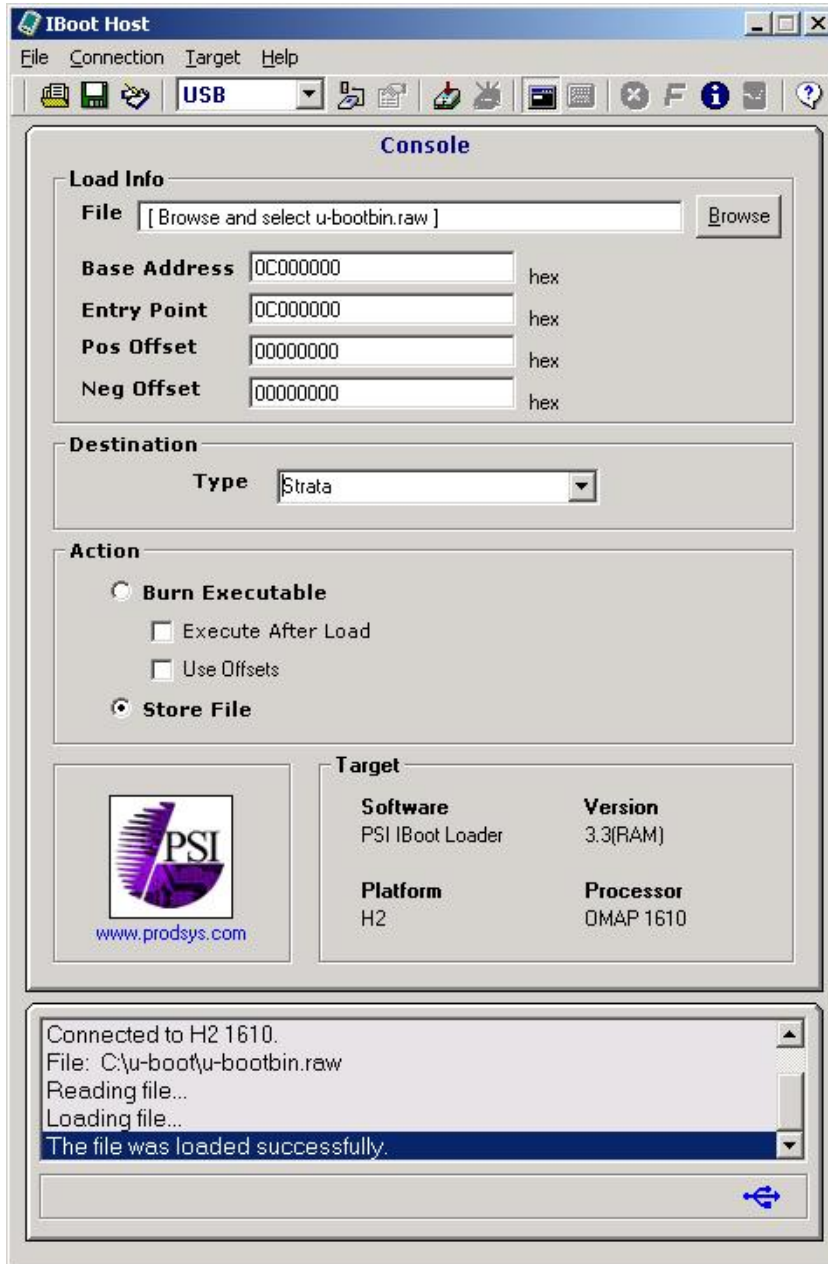
The host provides the file `"/tftpboot/plimage"` as boot image for the target.

9 Flashing U-Boot

To flash U-Boot on the OMAP1610 SDP, IBoot Host and IBoot Console software must be used. IBoot is released by Productivity Systems, Inc., and is shipped with every OMAP1610 SDP. The process described here might change if the IBoot software changes.

1. Install IBoot Host version 2.3 provided on the 'OMAP16xx GSM/GPRS Software Development Platform' Product Support CD.
2. From the CD, download and unzip IBoot Console Sources version 3.3.
3. Switch ARM BOOT = ON and power on the board.
4. Start CCS and load
C:\PSI\DEV\ConsoleApps\IBoot_Console\build\H2_1610\Debug_h16\IBootLoader_H16.out
5. Start Hyperterminal with the following settings (57600-8-N-1).

6. Press F5 to execute the program from CCS. Now you should see IBoot prompt on Hyperterminal.
7. Connect the USB cable from the PC to the SDP. Windows will ask you for the USB driver .inf file. Browse and provide the InnovatorUSB.Inf file included with the IBoot Host software setup package. If you have extracted the setup package in the default location, you may find the .inf file in C:\PSI\IBoot Host Rev2.3\IBoot Host Rev2.3\IBoot Host Driver\
8. On the IBoot prompt type "usb".
9. Now run IBoot host. By default it will be in
C:\PSI\IBOOT_HOST_1610\IBOOT_HOST_1610.exe
10. Click on *connect*. If connection fails, make sure the cable is connected properly and drivers are installed.
11. Rename the u-boot.bin file (generated after building U-boot) as something with a .raw extension, e.g. u-bootbin.raw. This helps IBoot Host to identify that the file is in raw binary format.
12. Fill in the fields of IBoot Host as shown below:



13. Click on *load*. This should load u-bootbin.raw to flash.
14. Now close CCS and power OFF the board.
15. Change the dip switch ARM BOOT to OFF, set HyperTerminal settings as (115200-8-N-1) and power ON the board. You should see U-Boot prompt on HyperTerminal.

10 Executing U-Boot

To run U-Boot from flash, the switch settings should be changed as described in section 5 (ARM BOOT = OFF). Connect a serial null-modem cable from the SDP to the PC, set HyperTerminal settings as (115200-8-N-1) and power ON the board.

WARNING:

When U-Boot executes you may see a message

```
*** Warning - bad CRC, using default environment
```

This is harmless and will go away as soon as you have initialized and saved the *environment* variables. This indicates that the environment variables are not stored in flash. After the environment variables are saved using **saveenv** command, this warning will disappear.

When the U-Boot executes, a message similar to this should be displayed on the HyperTerminal.

```
U-Boot 1.0.0 (Feb 16 2004 - 11:51:41)

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 #
```

U-Boot is now ready to accept commands. The easiest way to see the commands supported is to type 'help' on the command prompt, which results in a screen like this.

```
OMAP1610 H2 # help
?    - alias for 'help'
autoscr - run script from memory
base   - print or set address offset
bdinfo - print Board Info structure
boot   - boot default, i.e., run 'bootcmd'
bootd  - boot default, i.e., run 'bootcmd'
bootm  - boot application image from memory
```

bootp - boot image via network using BootP/TFTP protocol

bootd - boot default, i.e., run 'bootcmd'

cmp - memory compare

coninfo - print console devices and information

cp - memory copy

crc32 - checksum calculation

dhcp - invoke DHCP client to obtain IP/boot params

echo - echo args to console

erase - erase FLASH memory

flinfo - print FLASH memory information

go - start application at address 'addr'

help - print online help

imls - list all images found in flash

loadb - load binary file over serial line (kermit mode)

loads - load S-Record file over serial line

loop - infinite loop on address range

md - memory display

mm - memory modify (auto-incrementing)

mtest - simple RAM test

mw - memory write (fill)

nm - memory modify (constant address)

printenv- print environment variables

protect - enable or disable FLASH write protection

rarpboot- boot image via network using RARP/TFTP protocol

reset - Perform RESET of the CPU

run - run commands in an environment variable

saveenv - save environment variables to persistent storage

setenv - set environment variables

```
sleep - delay execution for some time
tftpboot- boot image via network using TFTP protocol
version - print monitor version
OMAP1610 H2 #
```

Help for a particular command can be requested by providing, "Help [command] ". One example is shown below.

```
OMAP1610 H2 # help md
md [.b, .w, .l] address [# of objects]
- memory display
```

The help here is self-explanatory and will not be explained comprehensively in this document.

11 Building a Linux Kernel Image to download from U-Boot

Please refer to *Building Linux for the OMAP™ Software Development Platform* [2] for building the Linux kernel. After the kernel is built, follow these steps to make the Linux kernel image suitable for download with U-Boot.

```
[root@localhost]# cd linux-x.y.z
[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
```

The mkimage tool comes with U-Boot and is placed in u-boot-x.y.z/tools directory.

This will create on ulmage.cc file, which can be downloaded and executed using U-Boot.

12 Downloading a Kernel Image with U-Boot

By default, the kernel image is downloaded to 0x10000000, which is the starting of SDRAM for the OMAP16xx SDP. But the user can use any address in SDRAM to load the image.

12.1 Serial Download

For serial download use *loadb* command

```
OMAP1610 H2 # help loadb
loadb [ off ] [ baud ]
- load binary file over serial line with offset 'off' and baudrate 'baud'
```

```
OMAP1610 H2 #
```

Serial download uses kermit protocol

```
OMAP1610 H2 # loadb
## Ready for binary (kermit) download to 0x10000000 at 115200 bps...
```

For Windows host

Use Hyperterminal

- a) Select Transfer->Send File
- b) Set protocol as Kermit
- c) Choose the "ulmage.cc" file (created in the preceding section) and press send.

12.2 TFTP Download

TFTP download is a very fast way to download the kernel images. *tftpboot* command has the following syntax.

```
OMAP1610 H2 # help tftpboot
tftpboot [loadAddress] [bootfilename]
```

An example of tftp download is given below

```
OMAP1610 H2 # tftpboot 0x10000000 "ulmage.cc"
TFTP from server 157.87.82.48; our IP address is 157.87.82.49
Filename 'ulmage.cc'.
Load address: 0x10000000
Loading: #####
          #####
          ###
done
Bytes transferred = 679196 (a5d1c hex)
OMAP1610 H2 #
```

12.3 Verifying the Download

After downloading the kernel image to SDRAM, verify that the image is downloaded properly using the *imi* command.

```
OMAP1610 H2 # imi 0x10000000

## Checking Image at 10000000 ...
Image Name: Linux Kernel Image
Image Type: ARM Linux Kernel Image (gzip compressed)
Data Size: 679132 Bytes = 663.2 kB
Load Address: 10c08000
Entry Point: 10c08000
Verifying Checksum ... OK

OMAP1610 H2 #
```

After downloading the kernel Image using serial or tftpboot to SDRAM, the kernel image can be stored to flash for permanent storage. Sections of flash where the kernel has to be stored must be erased before saving the kernel image. An example of storing the image is shown below:

```
OMAP1610 H2 # erase 1:8-13
Erase Flash Sectors 8-13 in Bank # 1
Erasing sector 8 ... done
Erasing sector 9 ... done
Erasing sector 10 ... done
Erasing sector 11 ... done
Erasing sector 12 ... done
Erasing sector 13 ... done
OMAP1610 H2 # cp.b 0x10000000 0x100000 a5d1c
Copy to Flash... done

OMAP1610 H2 #
```

Please be aware that all the numeric values given are hex values, regardless of whether they are preceded by 0x or not. (a5d1c in the 'cp' command above is the image size). After saving the image please check whether it is copied properly using *imi* command.

NOTE: Information about the flash can be obtained using the *flinfo* command. This command shows the total number of sectors and starting address of each sector.

13 Booting Linux kernel from U-Boot

Once the kernel image is stored in SDRAM or Flash, the Linux kernel can be started using *bootm* command.

```
OMAP1610 H2 # bootm 0x10000000
## Booting image at 10000000 ...
   Image Name:   Linux Kernel Image
   Image Type:   ARM Linux Kernel Image (gzip compressed)
   Data Size:    679132 Bytes = 663.2 kB
   Load Address: 10c08000
   Entry Point:  10c08000
   Verifying Checksum ... OK
   Uncompressing Kernel Image ... OK

Starting kernel ...
```

14 Upgrading U-Boot

If U-Boot is already running on your board, you can use this easily to download another U-Boot image and install this in place of the existing image.

WARNING:

Before you can install the new image, you have to erase the current one. It is strongly recommended that you make sure that:

1. You have a backup of the old image that works
2. You know how to install an image on a virgin system (ref section 9)

To install a new Image of U-Boot, the new Image must be downloaded first to SDRAM using *tftpboot* or *loadb* and then stored to flash.

E.g.:

```
OMAP1610 H2 # loadb 0x10000000
## Ready for binary (kermit) download to 0x10000000 at 115200 bps...
```

Send the latest u-boot.bin file via Hyperterminal (this is the .bin file generated after building U-Boot, no conversion is required for this image)

U-Boot resides in the first sector of the flash. So to flash the new Image of U-Boot sector 0, of flash must be erased.

```
OMAP1610 H2 # protect off 1:0
Un-Protect Flash Sectors 0-0 in Bank # 1
OMAP1610 H2 # erase 1:0
Erase Flash Sectors 0 in Bank # 1
Erasing sector 8 ... done
OMAP1610 H2 #
```

To store the new U-Boot image

```
OMAP1610 H2 # cp.b 0x10000000 0x0 16000
done
OMAP1610 H2 #
```

The new Image of U-Boot is now stored in flash. Provide *reset* command or do a power cycle to test this Image.

15 References

1. 1610 H2-SAMPLE Quick Start Guide, provided on the 'OMAP16xx GSM/GPRS Software Development Platform' Product Support CD.
2. Building Linux for the OMAP™ Software Development Platform.
3. U-Boot manual maintained by Wolfgang Denk
<http://www.denx.de/wiki/bin/view/DULG/Manual>
4. U-Boot mailing list: u-boot-users@lists.sourceforge.net

Appendix A

Download Linux kernel using tftpboot on Corporate Network

WARNING:

Consent from the network administrator should be taken before trying this out.

On a corporate network, workarounds may be required to download using tftpboot since network administrator may not allow creation of a DHCP server for this purpose on a live network.

A possible solution is described here. On the network there will be a global tftp server where the required files can be stored. Set the **serverip** environment variable as the ipaddress of this server. For the board set some free static IP addresses (free IP addresses can be requested from the network administrators, be careful as not to give any random IP address since this will conflict if the IP address has been allotted to some other machine).

So **ipaddr** environment variable can be set to this static IP address. If any gateway is needed that also can be setup using environment variables.

```

OMAP1610 Innovator # setenv serverip 157.87.82.48
OMAP1610 Innovator # setenv ipaddr 157.87.82.49
OMAP1610 Innovator # tftpboot 0x10000000 "ulmage.cc"
TFTP from server 157.87.82.48; our IP address is 157.87.82.49
Filename 'ulmage.cc'.
Load address: 0x10000000
Loading: #####
          #####
          ###
done
Bytes transferred = 679196 (a5d1c hex)

```

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Mailing Address:
Texas Instruments
Post Office Box 655303
Dallas, Texas 75265