



Marvell® PXA510 D2Plug Computer

Software User Guide

Doc. No. MV-S800823-00 Rev. -
November 18, 2011

Document Classification: Public



Document Conventions

	Note: Provides related information or information of special importance.
	Caution: Indicates potential damage to hardware or software, or loss of data.
	Warning: Indicates a risk of personal injury.

Document Status

Doc Status: Released	Technical Publication: 1.0
----------------------	----------------------------

For more information, visit our website at: www.marvell.com

Disclaimer

No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying and recording, for any purpose, without the express written permission of Marvell. Marvell retains the right to make changes to this document at any time, without notice. Marvell makes no warranty of any kind, expressed or implied, with regard to any information contained in this document, including, but not limited to, the implied warranties of merchantability or fitness for any particular purpose. Further, Marvell does not warrant the accuracy or completeness of the information, text, graphics, or other items contained within this document.

Marvell products are not designed for use in life-support equipment or applications that would cause a life-threatening situation if any such products failed. Do not use Marvell products in these types of equipment or applications.

With respect to the products described herein, the user or recipient, in the absence of appropriate U.S. government authorization, agrees:

- 1) Not to re-export or release any such information consisting of technology, software or source code controlled for national security reasons by the U.S. Export Control Regulations ("EAR"), to a national of EAR Country Groups D:1 or E:2;
- 2) Not to export the direct product of such technology or such software, to EAR Country Groups D:1 or E:2, if such technology or software and direct products thereof are controlled for national security reasons by the EAR; and,
- 3) In the case of technology controlled for national security reasons under the EAR where the direct product of the technology is a complete plant or component of a plant, not to export to EAR Country Groups D:1 or E:2 the direct product of the plant or major component thereof, if such direct product is controlled for national security reasons by the EAR, or is subject to controls under the U.S. Munitions List ("USML").

At all times hereunder, the recipient of any such information agrees that they shall be deemed to have manually signed this document in connection with their receipt of any such information.

Copyright © 2011, Marvell International Ltd. All rights reserved. Marvell, the Marvell logo, Moving Forward Faster, Alaska, Fastwriter, Datacom Systems on Silicon, Libertas, Link Street, NetGX, PHYAdvantage, Prestera, Raising The Technology Bar, The Technology Within, Virtual Cable Tester, and Yukon are registered trademarks of Marvell. Ants, AnyVoltage, Discovery, DSP Switcher, Feroceon, GalNet, GalTis, Horizon, Marvell Makes It All Possible, RADLAN, UniMAC, and VCT are trademarks of Marvell. Intel XScale® is a trademark or registered trademark of Intel Corporation or its subsidiaries in the United States and other countries. All other trademarks are the property of their respective owners.

Table of Contents

1	D2Plug Display Setup	5
2	Compiling D2Plug U-Boot and LSP Binaries	6
2.1	Download Marvell Cross-compiler Toolchain	6
2.2	Setup Toolchain	6
2.3	Compile U-Boot Binaries	6
2.4	Compile Kernel ulmage.....	7
3	Booting from a USB Drive	9
4	Reflash eMMC	13
5	Reflash uSD	15
6	Update U-Boot	17
6.1	Updating U-Boot via Network.....	17
6.2	Update U-Boot via USB	17
7	D2Plug U-Boot Recovery with Marvell XDB Debugger	18
7.1	Equipment Needed	18
7.2	Procedure.....	18
8	Video Playback	33
8.1	Play through Desktop Video Player	33
8.2	Play through GST-Launch Utility	33
8.3	Programmatically through Gstreamer	33



1 D2Plug Display Setup

This chapter describes the U-Boot setup to enable/disable either the HDMI or VGA display on D2Plug.

D2Plug HDMI output is controlled by the “clcd.lcd0_enable” environment variable. D2Plug VGA output is controlled by the “clcd.lcd1_enable” environment variable.

By default, both HDMI and VGA outputs are enabled.

See this from the U-Boot prompt:

```
Marvell>> print bootargs
bootargs=console=ttyS0,115200
video=dovefb:lcd0:1920x1080-16@60,lcd1:1024x768-16@60-edid clcd.lcd0_enable=1
clcd.lcd1_enable=1 usb0Mode=host root=/dev/mmcblk0p2 rootwait
```

To enable LCD only:

```
Marvell>> set bootargs 'console=ttyS0,115200
video=dovefb:lcd0:1920x1080-16@60,lcd1:1024x768-16@60-edid clcd.lcd0_enable=1
clcd.lcd1_enable=0 usb0Mode=host root=/dev/mmcblk0p2 rootwait'
Marvell>> save
Marvell>> reset
```

To enable VGA only:

```
Marvell>> set bootargs 'console=ttyS0,115200
video=dovefb:lcd0:1920x1080-16@60,lcd1:1024x768-16@60-edid clcd.lcd0_enable=0
clcd.lcd1_enable=1 usb0Mode=host root=/dev/mmcblk0p2 rootwait'
Marvell>> save
Marvell>> reset
```

To enable both LCD and VGA:

```
Marvell>> set bootargs 'console=ttyS0,115200
video=dovefb:lcd0:1920x1080-16@60,lcd1:1024x768-16@60-edid clcd.lcd0_enable=1
clcd.lcd1_enable=1 usb0Mode=host root=/dev/mmcblk0p2 rootwait'
Marvell>> save
Marvell>> reset
```

To disable both LCD and VGA:

```
Marvell>> set bootargs 'console=ttyS0,115200
video=dovefb:lcd0:1920x1080-16@60,lcd1:1024x768-16@60-edid clcd.lcd0_enable=0
clcd.lcd1_enable=0 usb0Mode=host root=/dev/mmcblk0p2 rootwait'
Marvell>> save
Marvell>> reset
```

2 Compiling D2Plug U-Boot and LSP Binaries

This chapter describes how to cross-compile D2Plug U-Boot and LSP binaries.

2.1 Download Marvell Cross-compiler Toolchain

Download “arm-marvell-linux-gnueabi-vfp.tar.bz2” from
<http://www.plugincomputer.org/downloads/d2plug/>

2.2 Setup Toolchain

Examples below assume:

- Cross-compiler setup under /home/ubuntu/toolchains
- arm-marvell-linux-gnueabi-vfp.tar.bz2 is placed under /home/ubuntu/toolchains

```
ubuntu@ubuntu-laptop:~$ cd /home/ubuntu/toolchains
ubuntu@ubuntu-laptop:~$ tar -xvzf arm-marvell-linux-gnueabi-vfp.tar.bz2
```

Place Marvell cross-compiler in the path:

```
ubuntu@ubuntu-laptop:~$ export
PATH=/home/ubuntu/toolchains/arm-marvell-linux-gnueabi-vfp/bin:$PATH;
```

2.3 Compile U-Boot Binaries

Download U-Boot sources from:

http://www.plugincomputer.org/405/us/d2plug/uboot/d2plug_uboot_src_v0p4.tar.bz2

Example below assumes using /home/ubuntu/build/uboot to build uboot.

```
ubuntu@ubuntu-laptop:~$ cd /home/ubuntu/build/uboot
```

Untar the source:

```
ubuntu@ubuntu-laptop:~$ tar -xvzf d2plug_uboot_src_v0p4.tar.bz2
```

CD to source directory:

```
ubuntu@ubuntu-laptop:~$ cd d2plug-uboot
```

Setup ARCH and CROSS_COMPILE variables:

```
ubuntu@ubuntu-laptop:~$ export ARCH=arm
ubuntu@ubuntu-laptop:~$ export CROSS_COMPILE=arm-marvell-linux-gnueabi-
```

Verify Marvell cross-compiler version:

You should see output below:

```
ubuntu@ubuntu-laptop:~$ arm-marvell-linux-gnueabi-gcc --version
```

```
arm-marvell-linux-gnueabi-gcc (FSF GNU GCC branch-4.4.5. Marvell GCC 2010q4-113)
4.4.5 20100614 (prerelease)
```

```
Copyright (C) 2010 Free Software Foundation, Inc.
```

```
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

Make clean:

```
ubuntu@ubuntu-laptop:~$ make mrproper
```

Make configuration:

```
ubuntu@ubuntu-laptop:~$ make rd88ap510avng_config BOOTROM=1 SPIBOOT=1
DRAM=SAMSUNG_1G NAND=0
```

Make:

```
ubuntu@ubuntu-laptop:~$ make
```

Alternatively, run `./build_scripts/build_rd88ap510avng_spi_2CS_1Gb_gang.sh` to build.

Binaries are created under `/home/ubuntu/build/uboot`. Use `u-boot-rd88ap510avng_samsung_1g_spi.bin`.

2.4 Compile Kernel ulmage

Download LSP kernel sources from:

http://www.plugcomputer.org/405/us/d2plug/kernel/d2plug_lsp_src_v0p4.tar.bz2

Example below assumes

- Building U-boot under `/home/ubuntu/build/lsp`
- Using `/home/ubuntu/build/lsp` to build kernel

```
ubuntu@ubuntu-laptop:~$ cd /home/ubuntu/build/lsp
```

Untar the source:

```
ubuntu@ubuntu-laptop:~$ tar -xvjf d2plug_lsp_src_v0p4.tar.bz2
```

CD to source directory:

```
ubuntu@ubuntu-laptop:~$ cd d2plug-linux-2.6.32.y
```

Setup ARCH and CROSS_COMPILE variables:

```
ubuntu@ubuntu-laptop:~$ export ARCH=arm;
ubuntu@ubuntu-laptop:~$ export CROSS_COMPILE=arm-marvell-linux-gnueabi-
```

Verify Marvell cross-compiler version:

You should see output below:

```
ubuntu@ubuntu-laptop:~$ arm-marvell-linux-gnueabi-gcc --version

arm-marvell-linux-gnueabi-gcc (FSF GNU GCC branch-4.4.5. Marvell GCC 2010q4-113)
4.4.5 20100614 (prerelease)

Copyright (C) 2010 Free Software Foundation, Inc.

This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

Make clean:

```
ubuntu@ubuntu-laptop:~$ make mrproper
```

Make configuration:

```
ubuntu@ubuntu-laptop:~$ make dove_d2plug_defconfig
```

Make:

```
ubuntu@ubuntu-laptop:~$ make uImage
```

Find created ulmage under /home/ubuntu/build/lsp/arch/arm/boot/

3

Booting from a USB Drive

This chapter describes how to create a USB drive and boot D2Plug from it. Follow instructions below on a Linux PC or D2Plug.

Insert USB drive. Setup USB drive with two partitions: Set Partition 1 size of 100MB and remaining space for Partition 2. Format Partition 1 as as “vfat” and partition 2 as “ext4”.

Example below assumes:

- Linux sees USB drive as /dev/sdc
- A 8GB USB drive is used

Find out what storage devices are there:

```
Ubuntu@D2Plug:~$ sudo fdisk -l
Ubuntu@D2Plug:~$ sudo mount
```

Unmount them if they are mounted:

```
Ubuntu@D2Plug:~$ sudo umount /dev/sdc1
Ubuntu@D2Plug:~$ sudo umount /dev/sdc2
```

Again, we assume the USB drive is /dev/sdc.

CAUTION: Operations below will erase all /dev/sdc contents.

```
Ubuntu@D2Plug:~$ sudo fdisk /dev/sdc
```

```
WARNING: DOS-compatible mode is deprecated. Marvell recommends
switching off the mode (command 'c') and changing display units to
sectors (command 'u').
```

```
Command (m for help): m
```

```
Command action
```

- a toggle a bootable flag
- b edit bsd disklabel
- c toggle the dos compatibility flag
- d delete a partition
- l list known partition types
- m print this menu
- n add a new partition
- o create a new empty DOS partition table

p print the partition table
q quit without saving changes
s create a new empty Sun disklabel
t change a partition's system id
u change display/entry units
v verify the partition table
w write table to disk and exit
x extra functionality (experts only)

Type 'm' to see help screen:

m

Type 'p' to see existing partitions:

p

If partition(s) already exist, erase with 'd':

d

Create the 2 partitions:

n

p

1

<CR>

+100M

n

p

2

<CR>

<CR>

w

Verify the partitions were created correctly with "sudo fdisk -l". You should see the following:

```
ubuntu@D2Plug:~$ sudo fdisk -l
Disk /dev/sdc: 8004 MB, 8004304896 bytes
35 heads, 21 sectors/track, 21269 cylinders
Units = cylinders of 735 * 512 = 376320 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00000000
```

Device	Boot	Start	End	Blocks	Id	System
/dev/sdc1		1	280	102889+	83	Linux
/dev/sdc2		281	21269	7713457+	83	Linux

Format partition 1 as "vfat":

```
Ubuntu@D2Plug:~$ sudo mkfs.vfat /dev/sdc1
```

Format partition 2 as "ext4":

```
Ubuntu@D2Plug:~$ sudo mkfs.ext4 /dev/sdc2
```

Verify partitions are formatted correctly:

```
Ubuntu@D2Plug:~$ sudo mount
```

You should see:

```
...  
/dev/sdc1 on /media/A4EC-6C0A type vfat  
(rw,nosuid,nodev,uhelper=udisks,uid=1000,gid=1000,shortname=mixed,dmask=0077,utf8  
=1,flush)  
/dev/sdc2 on /media/05b035c8-cb35-4b2d-afd0-8f08099a1f81 type ext4  
(rw,nosuid,nodev,uhelper=udisks)  
...
```



Note

If for some reason /dev/sdc1 and /dev/sdc2 are not automatically mounted, follow these steps to mount a partition manually:

```
# Mount /dev/sdc1 to /mnt
```

```
Ubuntu@D2Plug:~$ sudo mount /dev/sdc1 /mnt
```

```
# Unmount a partition by:
```

```
Ubuntu@D2Plug:~$ sync
```

```
Ubuntu@D2Plug:~$ cd
```

```
Ubuntu@D2Plug:~$ sudo umount /mnt;
```

Now copy files to the USB drive.

Outputs below assume:

- ulmage and rootfs are found under /home/ubuntu of the system used to prepare the USB drive
- /dev/sdc1 is mounted on /media/A4EC-6C0A
- /dev/sdc2 is mounted on /media/05b035c8-cb35-4b2d-afd0-8f08099a1f81

Copy ulmage to where /dev/sdc1 is mounted.

```
Ubuntu@D2Plug:~$ sudo cp ~/ulmage.d2plug /media/A4EC-6C0A
```

Untar rootfs to where /dev/sdc2 is mounted.

This will take time as the system is unzipping a 700M+ compressed file and writing the files to the USB drive.

IMPORTANT: The numeric-owner flag ensures the proper file permissions are preserved and must be used.

```
Ubuntu@D2Plug:~$ cd /media/05b035c8-cb35-4b2d-afd0-8f08099a1f81
Ubuntu@D2Plug:~$ sudo tar --numeric-owner -xzf ~/d2plug-rootfs-20110903.tar.gz
```

Sync the USB drive and reboot:

```
Ubuntu@D2Plug:~$ sync
Ubuntu@D2Plug:~$ sudo reboot
```

Set up D2Plug to boot from USB drive.

Stop at U-Boot prompt and make sure you can read Partition 1 of USB drive.

Make sure you can read USB drive Partition 1:

```
Marvell>> usb start
Marvell>> fatls usb 2:1
```

You should see ulmage.d2plug:

```
Marvell>> fatls usb 2:1
    3118880    uimage.d2plug
```

Change U-Boot parameter to boot up from the USB drive:

Set bootcmd to read from USB drive Partition 1

```
Marvell>> set bootcmd 'usb start; fatload usb 2:1 2000000 uImage.d2plug; bootm'
```

Set bootargs to mount rootfs from USB drive Partition 2

Replace root=/dev/mmcblk0p2 to root=/dev/sdc2

```
Marvell>> set bootargs 'console=ttyS0,115200
video=dovefb:lcd0:1920x1080-16@60,lcd1:1024x768-16@60-edid clcd.lcd0_enable=1
clcd.lcd1_enable=1 usb0Mode=host root=/dev/sdc2 rootwait'
```

Save, reset and let it boot:

```
Marvell>> save
Marvell>> reset
```

4

Reflash eMMC

This chapter describes the steps to reflash D2Plug eMMC units.

First follow the steps in Chapter 3 to boot up from the USB drive.

Once the system boots up to Linux, login and use "sudo mount" to see how /dev/mmcblk0p are mounted.*

```
ubuntu@D2Plug:/$ sudo mount
...
/dev/mmcblk0p2 on /media/eMMC_fs type ext4 (rw,nosuid,nodev,uhelper=udisks)
/dev/mmcblk0p1 on /media/eSATA_krnl type vfat
(rw,nosuid,nodev,uhelper=udisks,uid=1000,gid=1000,shortname=mixed,dmask=0077,utf8
=1,flush)
/dev/sdc1 on /media/A4EC-6C0A type vfat
(rw,nosuid,nodev,uhelper=udisks,uid=1000,gid=1000,shortname=mixed,dmask=0077,utf8
=1,flush)
...
```

Steps below assume "ulmage.d2plug" and "d2plug-rootfs-20110903.tar.gz" are copied to /home/ubuntu/ directory on the USB drive where the system boots off.

First umount /dev/mmcblk devices:*

```
ubuntu@D2Plug:~$ sudo umount /dev/mmcblk0p1
ubuntu@D2Plug:~$ sudo umount /dev/mmcblk0p2
```

Format /dev/mmcblk0p1 as "vfat"

```
Ubuntu@D2Plug:~$ sudo mkfs.vfat /dev/mmcblk0p1
```

Mount /dev/mmcblk0p1 to /mnt:

```
Ubuntu@D2Plug:~$ sudo mount /dev/mmcblk0p1 /mnt
```

Copy ulmage to /dev/mmcblk0p1:

```
ubuntu@D2Plug:~$ sudo cp uImage.d2plug /mnt
```

Format /dev/mmcblk0p2 as "ext4":

```
Ubuntu@D2Plug:~$ sudo mkfs.ext4 /dev/mmcblk0p2
```

Umount /mnt and mount /dev/mmcblk0p2 to /mnt:

```
Ubuntu@D2Plug:~$ sudo umount /mnt
```

```
Ubuntu@D2Plug:~$ sudo mount /dev/mmcblk0p2 /mnt
```

Untar rootfs to /dev/mmcblk0p2.

This will take time as the system is unzipping a 700M+ compressed file and writing the files to the USB drive.

IMPORTANT: The numeric-owner flag ensures the proper file permissions are preserved.

```
Ubuntu@D2Plug:~$ cd /mnt
Ubuntu@D2Plug:~$ sudo tar --numeric-owner -xzf ~/d2plug-rootfs-20110903.tar.gz
```

After tar completes, sync and reboot:

```
Ubuntu@D2Plug:~$ sync
Ubuntu@D2Plug:~$ sudo reboot
```

Stop at U-Boot prompt and make sure you can read Partition 1 of eMMC.

Make sure you can read “ulmage.d2plug” copied to /dev/mmcblk0p1.

```
Marvell>> mmcinfo;
Marvell>> fatls mmc 0:1
    3118880    uimage.d2plug
```

```
1 file(s), 0 dir(s)
```

You should see ulmage.d2plug.

Change U-Boot parameter to boot up from eMMC.

Set bootcmd to read ulmage from eMMC Partition 1:

```
Marvell>> set bootcmd 'mmcinfo; fatload mmc 0 0x2000000 uImage.d2plug; bootm
0x2000000'
```

Set bootargs to mount root file system from eMMC Partition 2.

Replace root=/dev/sdc2 with root=/dev/mmcblk0p2:

```
Marvell>> set bootargs 'console=ttyS0,115200
video=dovefb:lcd0:1920x1080-16@60,lcd1:1024x768-16@60-edid clcd.lcd0_enable=1
clcd.lcd1_enable=1 usb0Mode=host root=/dev/mmcblk0p2 rootwait'
```

Save, reset, and let it boot:

```
Marvell>> save
Marvell>> reset
```

5 Reflash uSD

This chapter describes steps on how to reflash D2Plug uSD units. Some early D2Plug units use internal uSD cards for non-volatile storage. You can also use a similar setup and reflash procedures for booting off an external SD card.

First follow the steps in Chapter 3 to setup and boot from the USB drive.

Once system boots up to Linux, login and use "sudo mount" to see how the uSD partitions (/dev/sda) are mounted.*

```
ubuntu@D2Plug:/$ sudo mount
...
/dev/sda2 on /media/266ec0df-13a2-4fcc-8bb3-8be930be8f1d type ext4
(rw,nosuid,nodev,uhelper=udisks)
/dev/sda1 on /media/USD_krnl type vfat
(rw,nosuid,nodev,uhelper=udisks,uid=1000,gid=1000,shortname=mixed,dmask=0077,utf8
=1,flush)
/dev/sdc1 on /media/A4EC-6C0A type vfat
(rw,nosuid,nodev,uhelper=udisks,uid=1000,gid=1000,shortname=mixed,dmask=0077,utf8
=1,flush)
...
```

Steps below assume "ulmage.d2plug" and "d2plug-rootfs-20110903.tar.gz" are copied to /home/ubuntu/ directory on the USB drive where the system boots off.

First umount /dev/sda devices:*

```
ubuntu@D2Plug:~$ sudo umount /dev/sda1
ubuntu@D2Plug:~$ sudo umount /dev/sda2
```

Format /dev/sda1 as "vfat":

```
Ubuntu@D2Plug:~$ sudo mkfs.vfat /dev/sda1
```

Mount /dev/sda1 to /mnt:

```
Ubuntu@D2Plug:~$ sudo mount /dev/sda1 /mnt
```

Copy ulmage to /dev/sda1:

```
ubuntu@D2Plug:~$ sudo cp uImage.d2plug /mnt
```

Format /dev/sda2 as "ext4":

```
Ubuntu@D2Plug:~$ sudo mkfs.ext4 /dev/sda2
```

Umount /mnt and mount /dev/sda2 to /mnt:

```
Ubuntu@D2Plug:~$ sudo umount /mnt
Ubuntu@D2Plug:~$ sudo mount /dev/sda2 /mnt
```

Untar rootfs to /dev/sda2:

This will take time as the system is unzipping a 700M+ compressed file and writing the files to the USB drive.

IMPORTANT: The numeric-owner flag ensures the proper file permissions are preserved.

```
Ubuntu@D2Plug:~$ cd /mnt
Ubuntu@D2Plug:~$ sudo tar --numeric-owner -xzf ~/d2plug-rootfs-20110903.tar.gz
```

After tar completes, Sync and reboot:

```
Ubuntu@D2Plug:~$ sync
Ubuntu@D2Plug:~$ sudo reboot
```

Stop at uboot prompt and make sure you can read Partition 1 of uSD.

Make sure you can read "ulmage.d2plug" copied to /dev/sda1.

```
Marvell>> usb start;
Marvell>> fatls usb 0:1
    3118880    uimage.d2plug

1 file(s), 0 dir(s)
```

You should see ulmage.d2plug.

Change U-Boot parameter to boot up from uSD.

Set bootcmd to read ulmage from uSD Partition 1:

```
Marvell>> set bootcmd 'usb start; fatload usb 0:1 0x2000000 uImage.d2plug; bootm
0x2000000'
```

Set bootargs to mount root file system from uSD Partition 2.

Replace root=/dev/sdc2 with root=/dev/sda2:

```
Marvell>> set bootargs 'console=ttyS0,115200
video=dovefb:lcd0:1920x1080-16@60,lcd1:1024x768-16@60-edid clcd.lcd0_enable=1
clcd.lcd1_enable=1 usb0Mode=host root=/dev/sda2 rootwait'
```

Save, reset, and let it boot:

```
Marvell>> save
Marvell>> reset
```

6 Update U-Boot

This chapter describes how to update the U-Boot firmware on the D2Plug device. U-Boot firmware can be updated via the network or the USB drive. See Section 2 of this guide for building D2Plug U-Boot binaries.

6.1 Updating U-Boot via Network

1. Set up a tftp server and place “u-boot-rd88ap0510avng_samsung_1g_spi.bin” under the tftpboot directory.
2. Connect ethernet cable to D2plug.
3. From the U-Boot console:

```
Marvell>> set ipaddr <ip_addr-of-d2plug>
Marvell>> set serverip <ip-addr-of-tftp-server>
Marvell>> burl u-boot-rd88ap510avng_samsung_1g_spi.bin
```

When prompted whether to “override Env parameters to default?”, decide accordingly.



Note

A free tftpd32 server is available for Windows hosts.

6.2 Update U-Boot via USB

1. Copy “u-boot-rd88ap510avng_samsung_1g_spi.bin” to the USB drive.
2. From the U-Boot console:

```
Marvell>> usb start
Marvell>> fatload usb 0 0x900000 u-boot-rd88ap510avng_samsung_1g_spi.bin
Marvell>> sf probe 0
```

```
Marvell>> sf protect off
Marvell>> sf erase 0x0 0xD0000
Marvell>> sf write 0x900000 0x0 0xD0000
Marvell>> reset
```

7 D2Plug U-Boot Recovery with Marvell XDB Debugger

7.1 Equipment Needed

- Marvell® XDB JTAG debugger
- Marvell® XDB JTAG debugger software
- D2Plug JTAG adapter board (available from GTI)

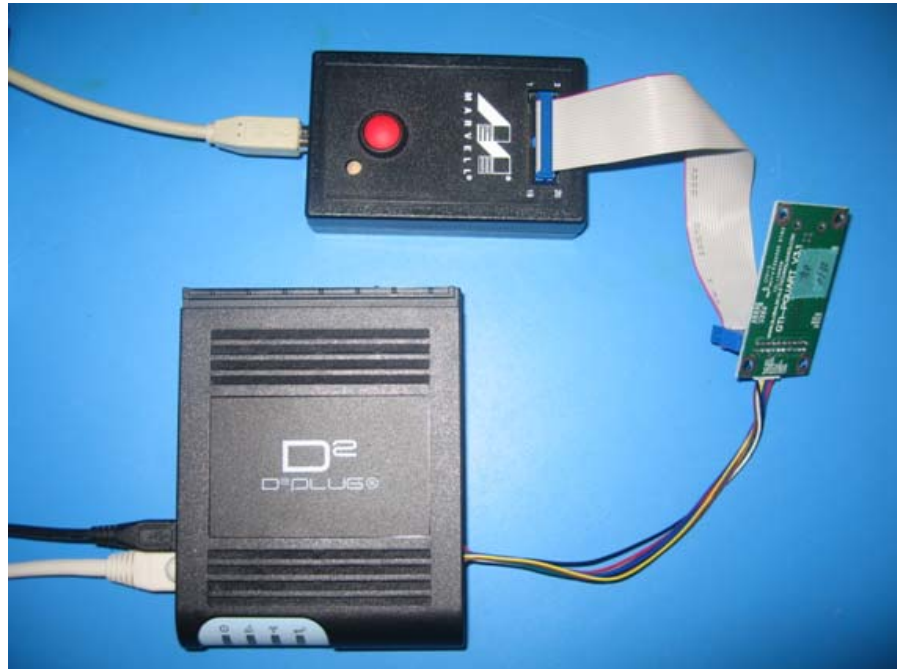
Figure 1: Marvell® XDB Adapter with GTI JTAG Adapter Board



7.2 Procedure

1. Connect USB serial cable to D2Plug console port.
2. Run Console program, such as TeraTerm on the PC with the following settings:
 - a) Baudrate: 115200
 - b) Data: 8 bit
 - c) Parity: None
 - d) Stop: 1 bit
 - e) Flow Control: None
3. Connect the XDB Debugger to the D2Plug JTAG port. Do not connect the XDB USB cable to the PC yet.

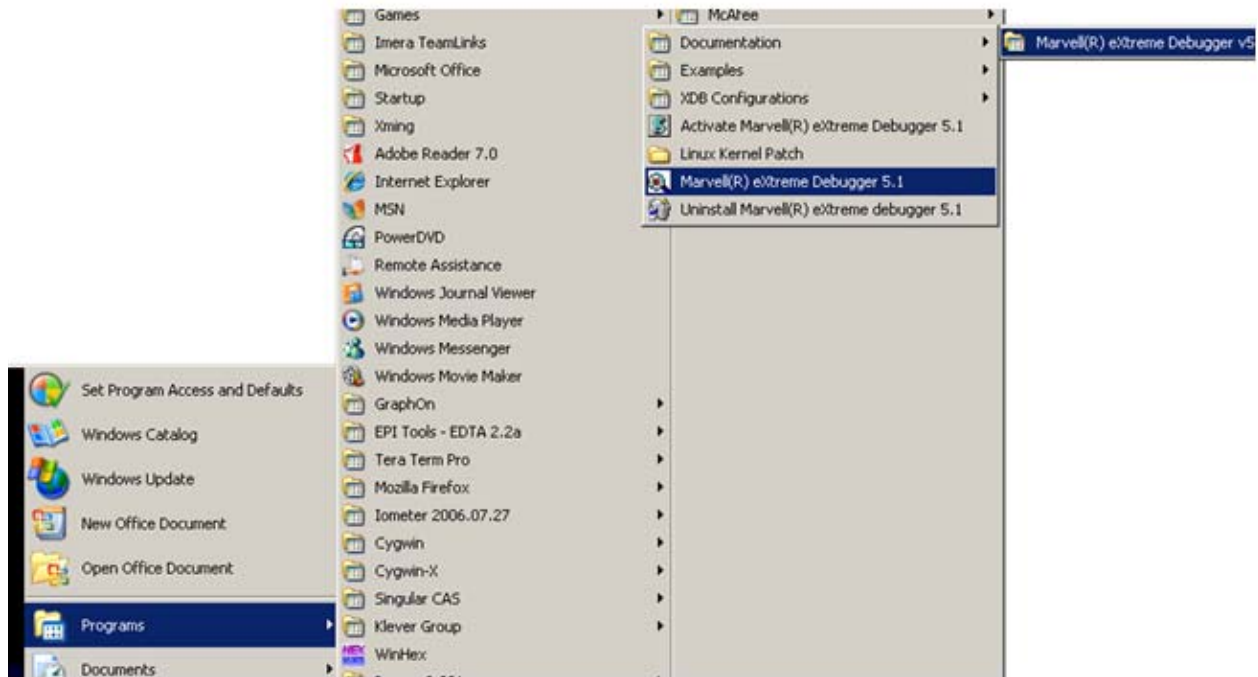
Figure 2: XDB-D2Plug-GTI JTAG Board Connection



4. Connect the Ethernet cable to the D2Plug.
5. Power on the D2Plug. If U-Boot is corrupted, there will not be any output on the console.
6. Connect XDB USB cable to the PC. The XDB debugger light turns to a solid orange color.

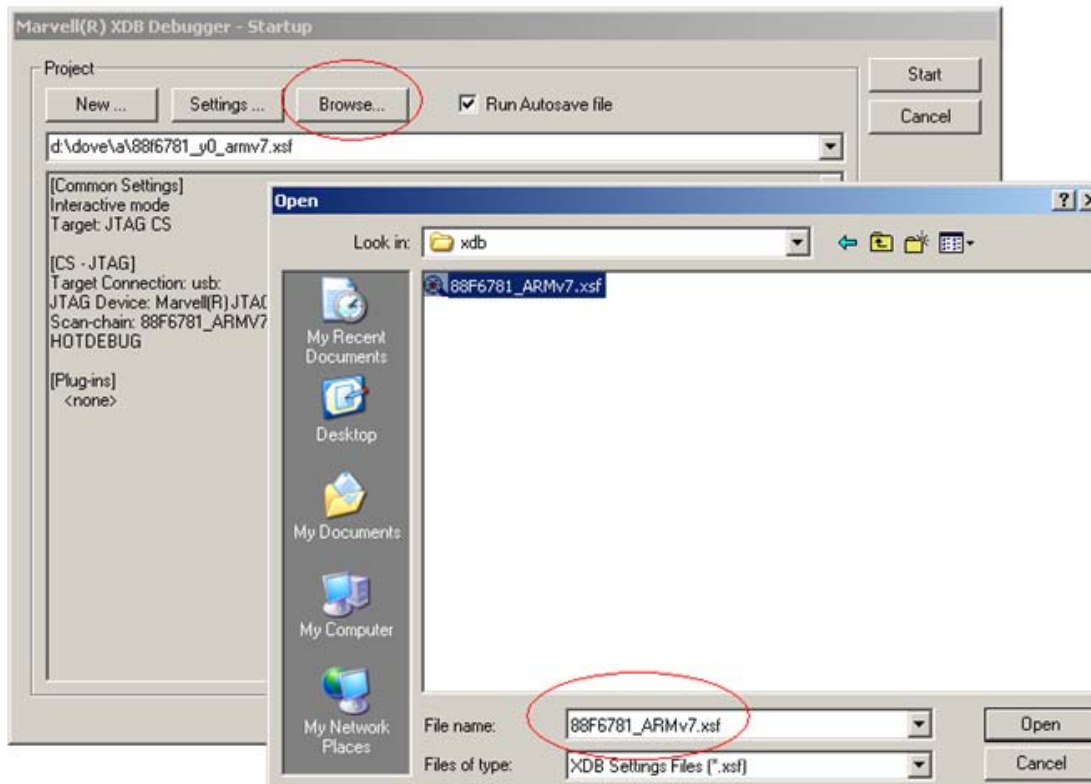
7. Run the Marvell eXtreme Debugger software on the PC (See Figure 3).

Figure 3: Screen Shot Showing eXtreme Debugger Application Access



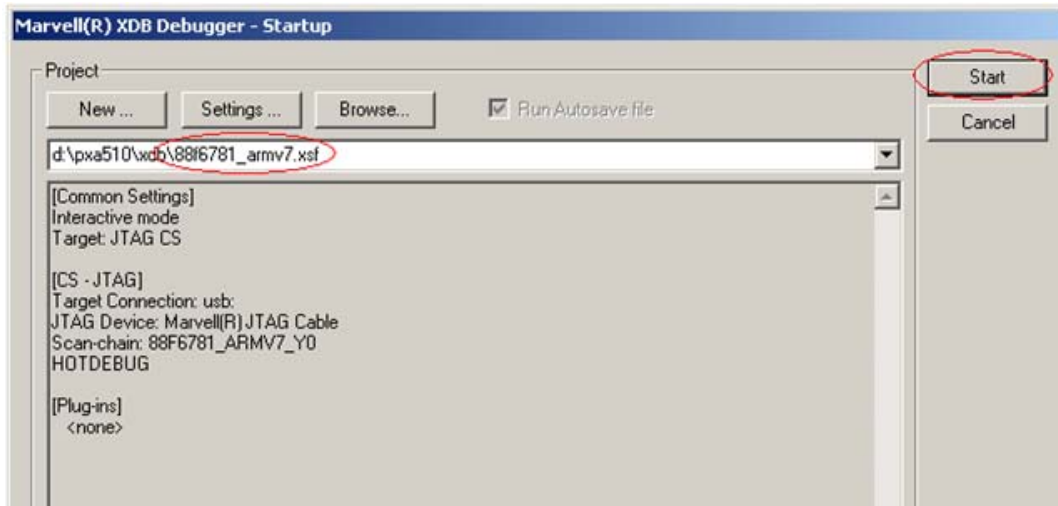
- Click on "Browse" to open "88F6781_ARMv7.xsf" (see Figure 4).

Figure 4: Accessing XDB Debugger



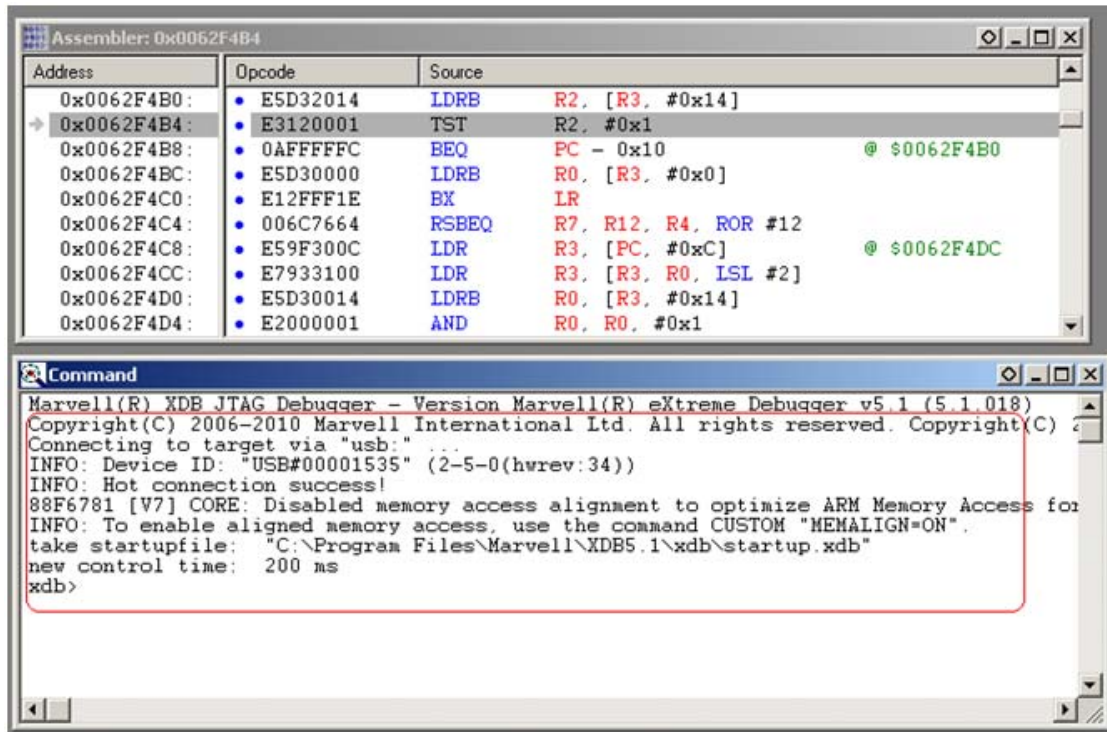
9. Click on "Start" (refer to Figure 5).

Figure 5: Screen Shot Showing "Start"



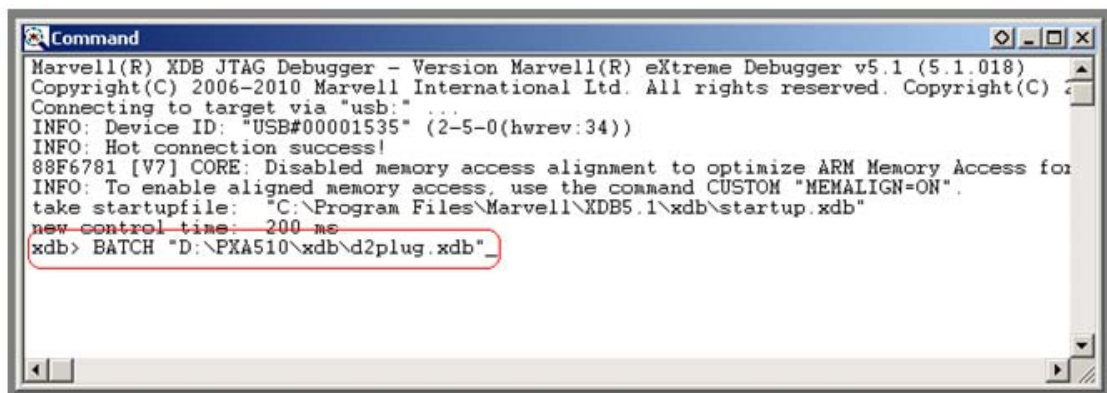
The XDB JTAG Debugger-Command window displays (see Figure 6):

Figure 6: XDB JTAG Debugger-Command Window



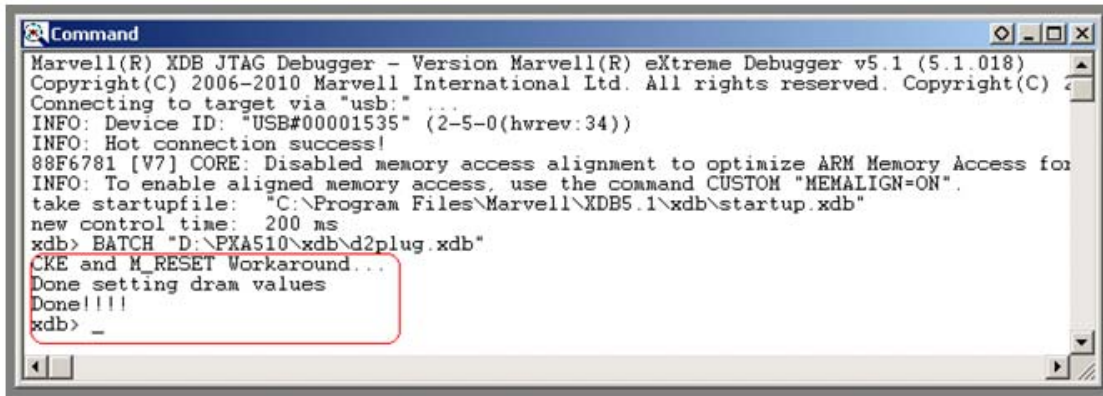
10. Run D2PlugXDB initialization script shown in the Command Window (see Figure 7).

Figure 7: Command Window showing Command Line Call of Batch File



The xdb> prompt displays after the script has finished execution, as shown in Figure 8.

Figure 8: Command Line Showing xdb> Prompt

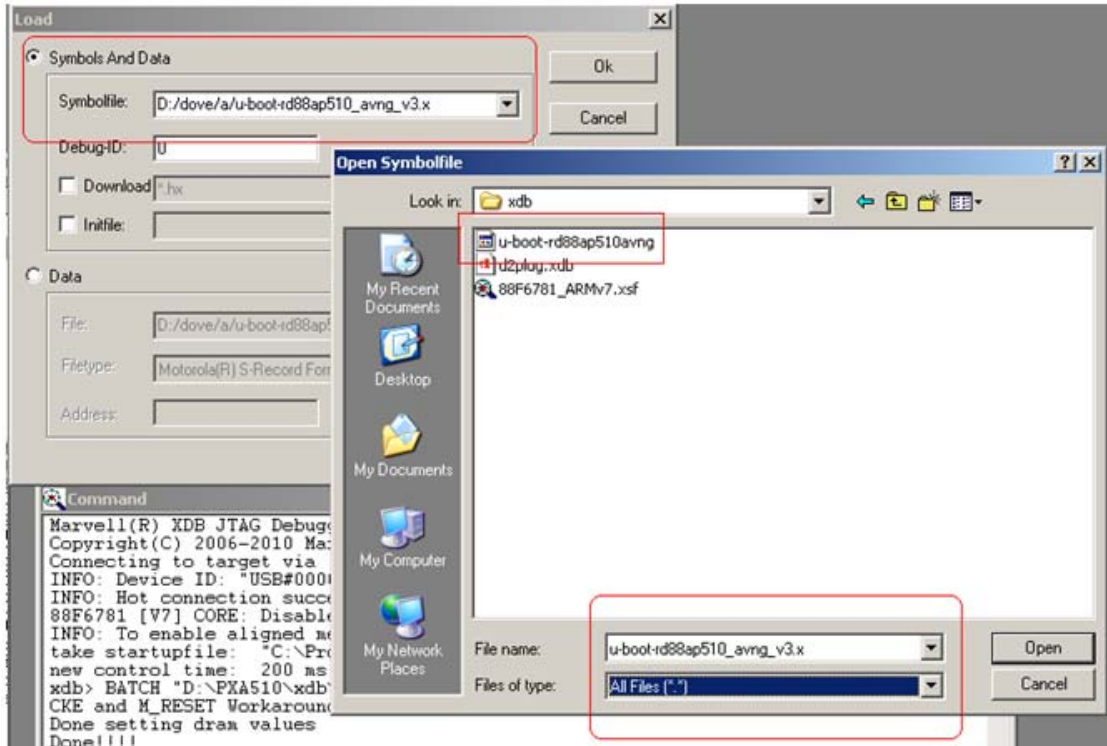


```
Command
Marvell(R) XDB JTAG Debugger - Version Marvell(R) eXtreme Debugger v5.1 (5.1.018)
Copyright(C) 2006-2010 Marvell International Ltd. All rights reserved. Copyright(C)
Connecting to target via "usb:" ...
INFO: Device ID: "USB#00001535" (2-5-0(hwrev:34))
INFO: Hot connection success!
88F6781 [V7] CORE: Disabled memory access alignment to optimize ARM Memory Access for
INFO: To enable aligned memory access, use the command CUSTOM "MEMALIGN=ON".
take startupfile: "C:\Program Files\Marvell\XDB5.1\xdb\startup.xdb"
new control time: 200 ms
xdb> BATCH "D:\PXAS10\xdb\d2plug.xdb"
CKE and M_RESET Workaround...
Done setting dram values
Done!!!!
xdb> _
```

11. Click on File, select "Load" to load U-Boot ELF file.

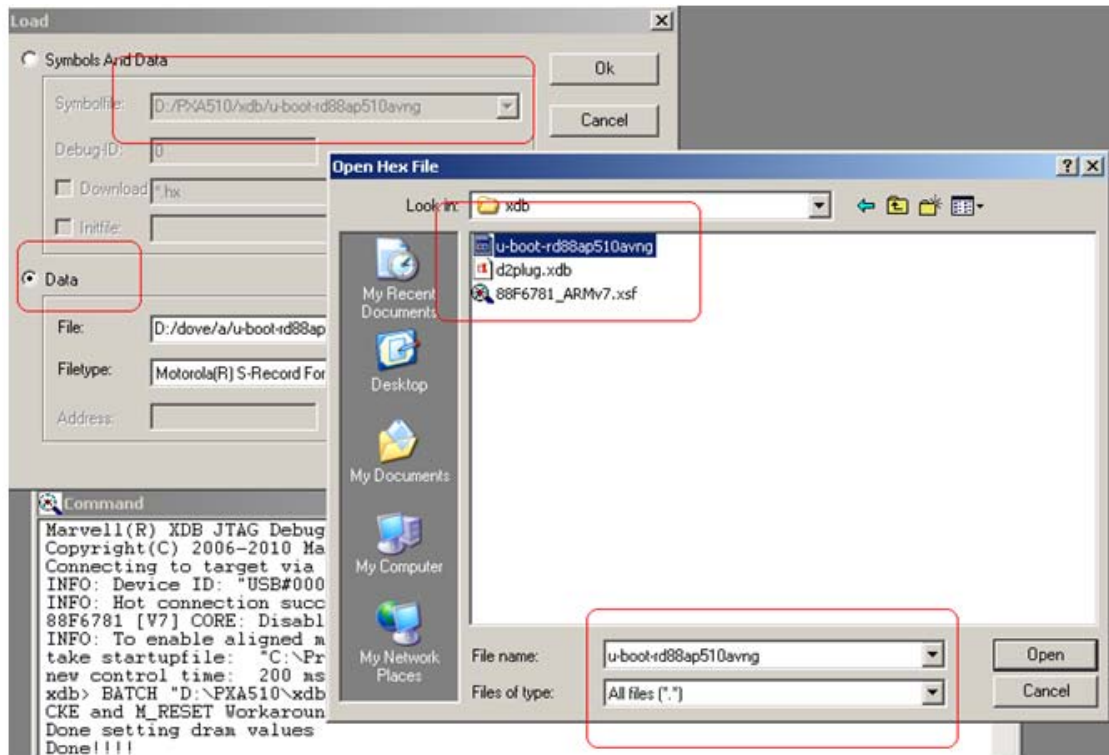
The U-Boot ELF is generated when D2Plug U-Boot binaries are compiled. Look for the file named “u-boot-rd88ap510avng” without any extension (approx. 3 MB in size). See Figure 9.

Figure 9: Location of the File “u-boot-rd88ap510avng”



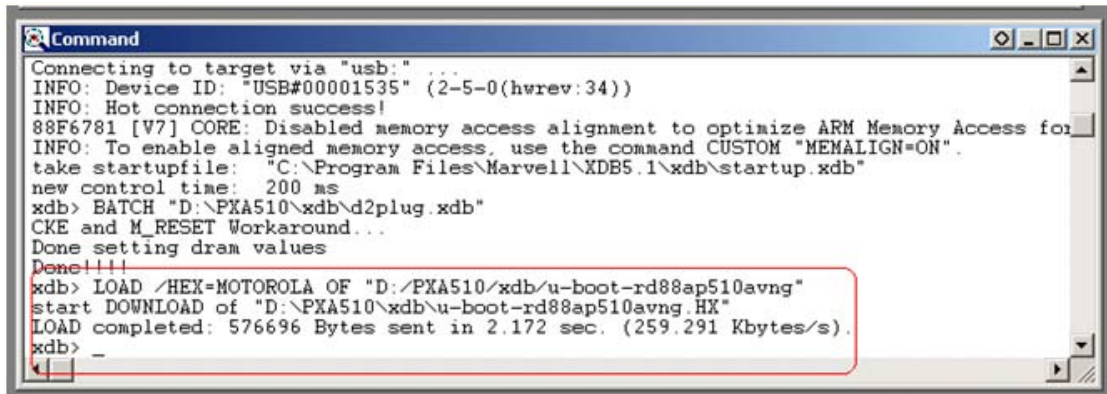
12. Repeat the steps to load the ELF file for Data, then click “OK” (see Figure 10).

Figure 10: Loading ELF file for “Data”



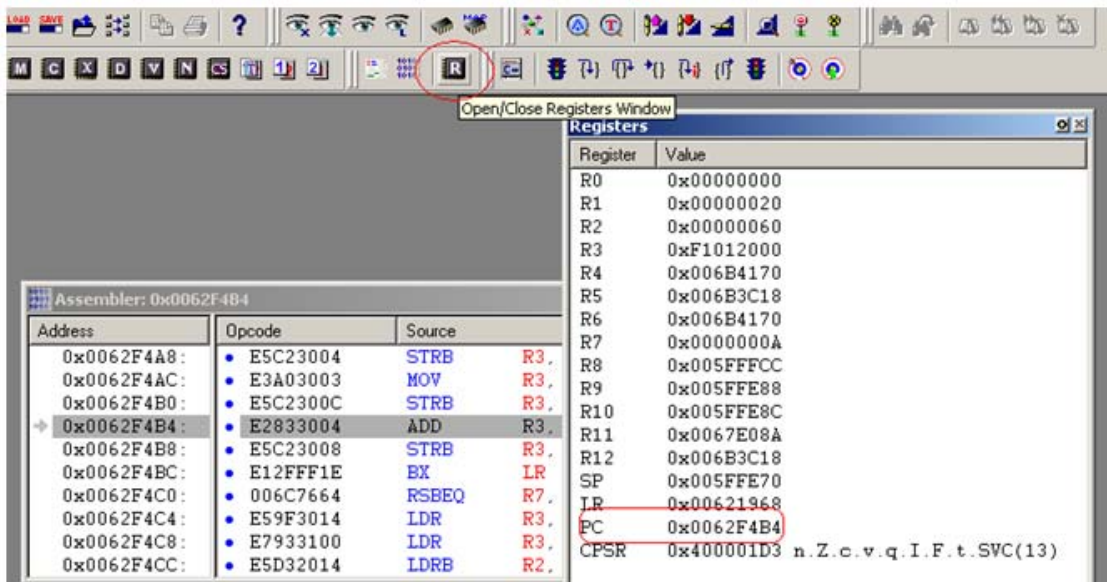
Command screen then indicates that the ELF file has loaded (see Figure 11).

Figure 11: Command Screen Indicating ELF File has Loaded



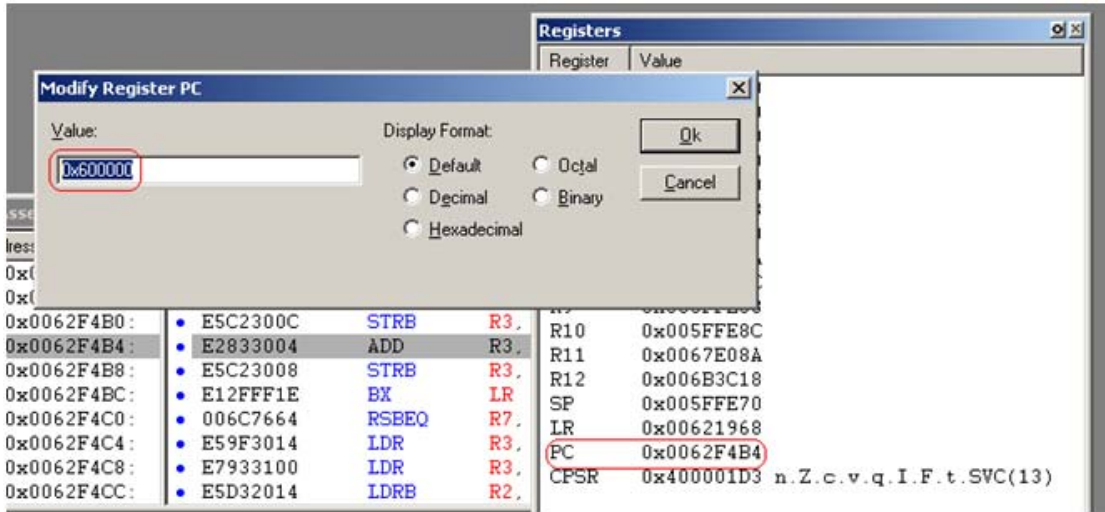
13. From the toolbar, click on the "R" icon to open the Register window (see Figure 12).

Figure 12: Screen Showing Selection of "Register" Window



14. Double-click on “PC” and in the “Modify Register PC” window, set the value to 0x600000 (see Figure 13).

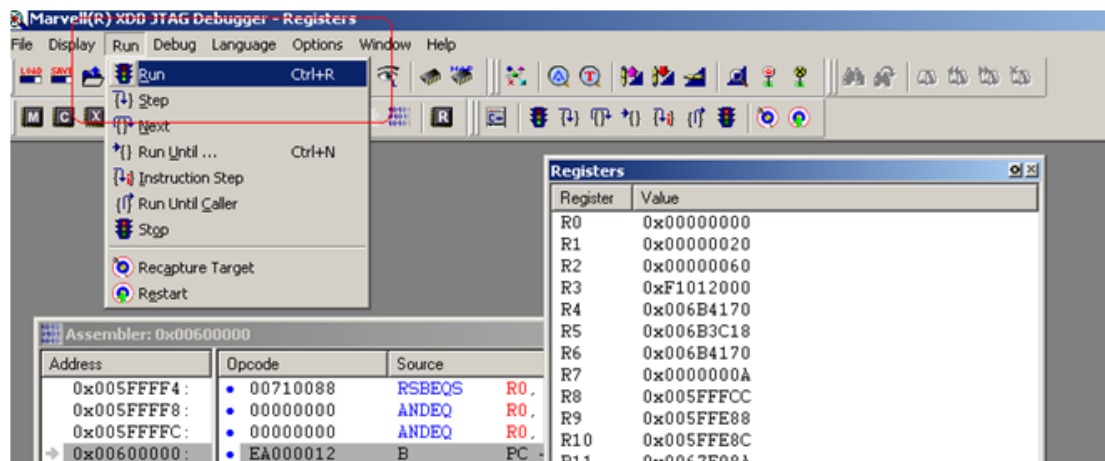
Figure 13: Screen Showing Modification of the PC Register



The PC is now set to 0x600000 and is ready to begin execution.

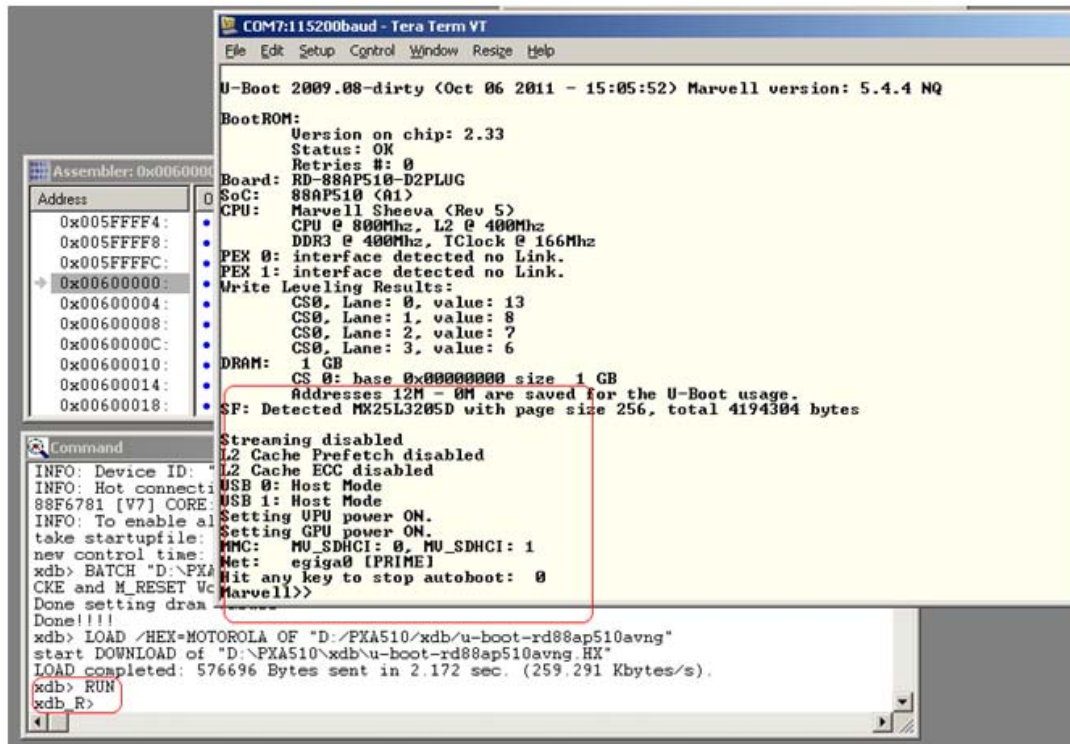
15. From the toolbar, click on “Run” and select Run (see Figure 14).

Figure 14: Screen Showing Selection of “Run”



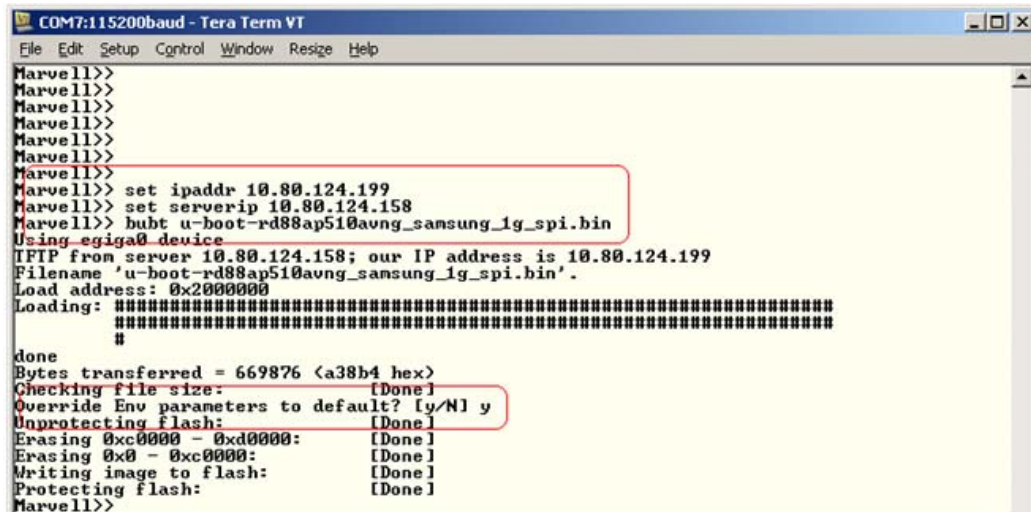
The TeraTerm window displays the U-Boot prompt on the console (see Figure 15).

Figure 15: U-Boot Prompt on the Console



16. From the console, press “Enter” to stop autoboot. Use “bubt” to write U-Boot to Flash (see Figure 16).

Figure 16: Screen Showing use of “bubt” to Write U-Boot to Flash



```
COM7:115200baud - Tera Term VT
File Edit Setup Control Window Resize Help
Marvell>>
Marvell>>
Marvell>>
Marvell>>
Marvell>>
Marvell>>
Marvell>>
Marvell>> set ipaddr 10.80.124.199
Marvell>> set serverip 10.80.124.158
Marvell>> bubt u-boot-rd88ap510avng_samsung_1g_spi.bin
Using egiqa0 device
TFTP from server 10.80.124.158; our IP address is 10.80.124.199
Filename 'u-boot-rd88ap510avng_samsung_1g_spi.bin'.
Load address: 0x2000000
Loading: #####
#
done
Bytes transferred = 669876 (a38b4 hex)
Checking file size: [Done]
Override Env parameters to default? [y/N] y
Unprotecting flash: [Done]
Erasing 0xc0000 - 0xd0000: [Done]
Erasing 0x0 - 0xc0000: [Done]
Writing image to flash: [Done]
Protecting flash: [Done]
Marvell>>
```

Recovery has been completed.

17. Quit the Marvell eXtreme Debugger program
18. Unplug the power to the D2Plug
19. Unplug the JTAG cable from the D2Plug

20. Power on the D2Plug. The serial console displays the following message (Figure 17):

Figure 17: U-Boot Recover Screen

```
Marvell
U-Boot
** LOADER **

U-Boot 2009.08-dirty <Oct 06 2011 - 15:05:52> Marvell version: 5.4.4 NQ
BootROM:
  Version on chip: 2.33
  Status: OK
  Retries #: 0
Board: RD-88AP510-D2PLUG
SoC: 88AP510 (A1)
CPU: Marvell Sheeva (Rev 5)
  CPU @ 800Mhz, L2 @ 400Mhz
  DDR3 @ 400Mhz, TClock @ 166Mhz
PEX 0: interface detected no Link.
PEX 1: interface detected no Link.
Write Leveling Results:
  CS0, Lane: 0, value: 13
  CS0, Lane: 1, value: 8
  CS0, Lane: 2, value: 6
  CS0, Lane: 3, value: 5
DRAM: 1 GB
  CS 0: base 0x00000000 size 1 GB
  Addresses 12M - 0M are saved for the U-Boot usage.
SF: Detected MX25L3205D with page size 256, total 4194304 bytes
*** Warning - bad CRC, using default environment

Running diagnostics ...

  DDR3 data bus test                PASSED
  DDR3 address bus test             PASSED
  DDR3 device test
```

Because the U-Boot environmental variables were reset, the system automatically runs a diagnostic check, which will require several minutes.

Once the diagnostics have completed, disable the diagnostics with the following U-Boot commands:

```
Marvell>> set run_diag no
Marvell>> save
Marvell>> reset
```

See Figure 18.

Figure 18: Disabling U-Boot Diagnostics

```
marvell>>
marvell>>
marvell>>
marvell>>
marvell>>
marvell>>
marvell>> set run_diag no
marvell>> save
Saving Environment to SPI Flash...
Inprotecting flash:           [Done]
Erasing 0xc0000 - 0xd0000:   [Done]
Writing to SPI flash:        [Done]
Protecting flash:            [Done]
marvell>> reset
```

After “reset”, U-Boot no longer runs the diagnostics automatically.

8 Video Playback

This chapter describes how to play a video on the D2Plug device. There are three options available: using the Ubuntu Desktop Video Player, with a GST-Launch utility from the terminal, or programmatically.

8.1 Play through Desktop Video Player

1. Copy video file to the D2plug /home/ubuntu/videos directory
2. Double-click the video file from the desktop

8.2 Play through GST-Launch Utility

1. Copy video file to the D2Plug /home/ubuntu/videos directory
2. Open a desktop terminal window
3. To launch video, enter the following in one line (replace “movie.mp4” with actual video filename):

```
Ubuntu@D2Plug:~$ gst-launch-0.10 playbin2
uri=file:///home/ubuntu/videos/movie.mp4 video-sink="bmmxvimagesink
device=0"
```

4. To capture the debug log in file “1.log”:

```
Ubuntu@D2Plug:~$ export GST_DEBUG=3
Ubuntu@D2Plug:~$ gst-launch0.10 playbin2
uri=file:///home/ubuntu/videos/movie.mp4 video-sink="bmmxvimagesink
device=0" 2>&1 | tee 1.log
```



Note

First issue this command if running from the serial console:

```
Ubuntu@D2Plug:~$ export DISPLAY=:0.0
```

There are three adapters for XVideo:

- Mixed Overlay Adapter (bmmxvimagesink device=0): Runs through both GPU and LCD overlays. This adapter gives the best results but is more taxing on system bandwidth.
- GPU Overlay Adapter (bmmxvimagesink device=1): Some issues with tearing.
- LCD Overlay Adapter (bmmxvimagesink device=2): Scaling down the video port too small may cause video to “garble.”

8.3 Programmatically through Gstreamer

The multimedia features of the D2Plug are accessible through the GStreamer (gst) framework. Documentation for GStreamer is available in the public domain.

Application Manual:

<http://gstreamer.freedesktop.org/data/doc/gstreamer/head/manual/html/index.html>

Plugin Manual:

<http://gstreamer.freedesktop.org/data/doc/gstreamer/head/pwg/html/index.html>