ARMADA™ 610 Tablet Reference Platform

for Android™ 2.3, Linux® Kernel 2.6.35

Software Release Notes

Doc. No. MV-S302040-00, Rev. C
July 19, 2011
CONFIDENTIAL
Document Classification: Proprietary Information

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	

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

Table of Contents

1 Overview

This software release package contains source code for the Marvell® ARMADA™ 610 Tablet Reference Platform for Android™ 2.3, Linux® kernel 2.6.35.

The release package includes:

- Prebuilt binaries – Use to flash the ARMADA 610 Tablet Reference Platform; these binaries are ready for immediate use.
- Source code – Customize and build the code to create new binaries.

THIS SOFTWARE IS PREPRODUCTION SOFTWARE. IT CANNOT BE USED "AS IS" FOR PRODUCTION SYSTEMS. NO WARRANTY, EXPRESSED OR IMPLIED, IS TO BE ASSOCIATED WITH THIS SOFTWARE AND THE USER ASSUMES ALL RISKS.

For the latest Marvell documentation and the latest Marvell software and hardware updates, contact your Marvell representative.

1.1 System Requirements

1.1.1 Hardware Requirements

This release requires the Marvell ARMADA 610 Tablet Reference Platform with A0/A1/A2 stepping of the ARMADA 610 processor and 1 GB of DDR3 memory.

All applicable Engineering Change Orders (ECOs) for a particular hardware revision must be applied to ensure that the software operates properly.

For the ECO level supported by this release and for ECO documentation, see Table 1.

Table 1: ECO Information for ARMADA 610 Tablet Reference Platform

ECO Level Requirement	Document	Document Number
Revision 1 • PB number: F00055-100 • ECO level: ECOs 1 through 22 (0x3FFFFF)	*Marvell® ARMADA™ 610 Tablet Reference Platform, Revision 1 Engineering Change Orders*	MV-S501271-00
Revision 2 • PB number: F00055-200 • ECO level: ECOs 1 through 12 (0xFFF)	*Marvell® ARMADA™ 610 Tablet Reference Platform, Revision 2 Engineering Change Orders*	MV-S501274-00
Revision 3 • PB number: F00055-300 • ECO level: ECOs 1 through 8 (0xFF)	*Marvell® ARMADA™ 610 Tablet Reference Platform, Revision 3 Engineering Change Orders*	MV-S501275-00
Revision 4 • PB number: F00055-400 • ECO level: ECO 9 (0b1001)	*Marvell® ARMADA™ 610 Tablet Reference Platform, Revision 4 Engineering Change Orders*	MV-S501286-00

The printed board (PB) number F00055 identifies the board as a Marvell ARMADA 610 Tablet Reference Platform. The -100, -200, -300, and -400 indicate revision 1, revision 2, revision 3, and revision 4, respectively, of the unpopulated board and schematics.

For the ECO level, an ECO label is applied to the board that encodes the numbers of the ECOs added to the board into a hexadecimal number. Refer to the appropriate Engineering Change Orders document for detailed ECO information.

1.1.2 Software Requirements

This release version requires:

- Host PC with operating system – Ubuntu® 10.04

Note Marvell routinely tests the Android build on Ubuntu 10.04. Marvell recommends using Ubuntu 10.04 and manually upgrading the Git to v1.7.1 or later. You can download the Git package from http://git-scm.com/download.

Note Follow the instructions on http://source.android.com/source/download.html to set up the Android build environment on Ubuntu Linux.

- Android 2.3, Linux kernel 2.6.35

1.2 Differences from the Previous Release (Beta 1)

Differences between the current release and the Beta 1 release follow:

- **BSP Differences**
  - Supports a larger audio buffer
  - Supports Vmeta™ profiling for saving power
  - Shows the kernel init logo on the LCD panel
- **Android Differences**
  - Provides some Android Compatibility Test Suite (CTS) fixes
  - Supports digital rights management (DRM) playback
  - Supports 8787 p66 firmware
  - Supports both trusted and non-trusted boots
  - Enables some skia graphics controller unit (GCU) optimizations for RGB 8888 or 565
  - Allows changing the TV resolution dynamically during HDMI™ playback
  - Enables rotation in HDMI mirror mode
- **Other Differences**
  - Provides DRM OpenMAX™ IL component update with bug fixes and enhancements
  - Provides JPEG decoder update with error handling enhancement
  - Upgrades the graphics controller (GC) libraries to version 0.8.0.3287p
  - Provides an integrated texture map fix for a mass production build
  - Fixes for many games that cannot run or that render error issues
  - Adds a stream valid check for the OpenGL® ES 2.0 build stream path
  - Refines the depth and 2D blending related code
  - Enables a GC silent reset
  - Adds an application frame rate limitation to save power, with the default off
  - Fixes the skia GCU pre-multiplied color blend error

- Refines the idle register access sequence to make sure that the GC is idle
- Fixes the blur/dim layer rendering to a black window issue
- Prevents a power off of the GC before entering early suspend as a workaround for the white screen issue
- Disables the user mode heap allocation
- Fixes the switch between suspend and resume modes in video playing that creates the wallpaper hang issue
- Refines the glFinish function to avoid the commission of GC commands when there is no drawing
- Adds a delay to clear the 2D path for user interface (UI) performance
- Fixes a small video blur or darkness during rotation, forcing the upload of a filter every time
- Fixes the garbage issue when exiting from the camera; refines the skip 2 frame logic
- Fixes the gal2D multi-thread test that causes the GUI hang issue

1.3 Differences for Previous Releases

Differences for previous releases follow:

- **BSP Differences**
  - Uses a new HDMI stack with kernel HDMI UIO driver support
  - Uses an LCD Display Serial Interface (DSI) 4 lane setting instead of a 2 lane setting
  - Supports the Extended File System, version 4 (ext4) filesystem, recovery mode, and fastboot mode
  - Enables the USB OTG switch to host mode automatically when the USB micro-A cable is connected
  - Unifies and enlarges the physical continuous memory (PMEM) buffer to 192 MB
  - Adds DLL update support in the Dynamic Voltage and Frequency Management (DVFM)
  - Moves the video DMA (VDMA) usage from the video layers to the graphic layers
  - Adds core 988 MHz/DDR 400 MHz production point support
  - Replaces Vmeta memory usage from Buffer Management Module (BMM) to PMEM
  - Adds the Marvell logo on the LCD panel during U-Boot initialization
  - Supports a FUSE read through the Wireless Trusted Module (WTM) image
  - Adds QCIF, CIF, and HVGA support in the Camera driver
  - Changes the default memory configuration from 1GB to 512 MB
- **Android Differences**
  - Implements a software upgrade and recovery
  - Uses an ext4 filesystem
  - Uses software decoders for thumbnail generation
  - Replaces the memory usage in Vmeta and the camera engine from BMM to PMEM
  - Adds the HDMI mirror mode and high-bandwidth digital content protection (HDCP) support
  - Supports both trusted and non-trusted boot
  - Supports fastboot
  - Supports all languages
  - Builds in the TTS language packs
  - Supports video rotation

- **Other Differences**
  - Upgrades the graphics controller (GC) libraries to version 0.8.0.2835
  - Fixes the OpenGL ES 2.0 UI NP2 texture does not draw correctly issue
  - Fixes the UnlockScreen hangs and Angrybird seasons corruption issue
  - Refines the depth mask setting and non-paged memory map/unmap logic
  - Enables 64 alignment unpack for GCU acceleration
  - Fixes gc860 alpha blending and software blur issue
  - Includes the Marvell Vmeta hardware abstract layer (HAL) upgrade to version 20110211
  - Upgrades Marvell Wireless Trusted Platform Tool package (WTPTP) to v3.2.19
  - Upgrades the Marvell WTM firmware to v2.1.6.
  - Supports the Marvell trusted boot flow
  - Supports the HDCP key protection

1.4 Platform Features

Table 2 lists the platform features for this version of the Marvell ARMADA 610 Tablet Reference Platform for Android 2.3 release package.

Table 2: Platform Features (Sheet 1 of 2)

Features		Support
General	Android Version	2.3
	Linux kernel	2.6.35
Power Management	Android power integration	Yes
	Battery information	No
	Suspend/Resume	Yes
Video Playback	Video output to HDMI™	Yes
	Video output optimized by overlay	Yes
	Simultaneous video output to HDMI and UI operation on LCD	Yes
	Simultaneous different video content playback on HDMI and LCD	Yes
	Video rotation	Yes
	Video output through graphics controller	No
Audio Playback	Headset switch detection	Yes
	Audio driver integration	Yes
	Audio to headset	Yes
	Audio to speaker	Yes
	Audio to HDMI	Yes
Audio Recording	AMR-NB encoding	Yes
	AAC encoding	Yes
	Sound recorder integration	Yes

Table 2: Platform Features (Sheet 2 of 2)

Features		Support
Video Recording	Camera stack integration	Yes
	Camera sensor tuning	Yes
Touch	Single touch	Yes
	Multiple touch	Yes
Bluetooth® technology	Bluetooth base stack	Yes
	Advanced Audio Distribution Profile (A2DP)	Yes
Sensors	Gravity sensor	No
	Light sensor	Yes
	Proximity sensor	Yes
Light	LCD backlight	Yes
	Keypad backlight	Yes
Alarm	Trigger alarm from Standby	No
Wireless	Wi-Fi® access without password	Yes
	Wi-Fi Protected Access (WPA)/WPA2	Yes
	Wi-Fi connection stress test	Yes
	3G USB Dongle	Yes
Multimedia	GStreamer	See Table 3, Supported Media Format and Codecs, on page 10
	OpenCore	
Video Output	Output through graphics controller	No
	Output through overlay	Yes
Tools	Android Debug Bridge (ADB) integration	Yes
	Marvell Code Performance Analyzer integration	Yes
Graphics	2D/3D graphics controller (GC860[1])	Yes
User Storage	SD card	Yes
	Internal storage partition	Yes
Boot Storage	eMMC	Yes
	SD	Yes
System Update	Fast boot protocol	Yes
	SD upgrade	Yes
Security	Wireless Trusted Platform Service Package (WTPSP)	Yes
	Wireless Trusted Module (WTM) Adapter for Linux Kernel Crypto Framework	Yes
	Optimized OpenSSL	Yes

1. GC860 refers to the Vivante Corporation GCCORE Graphics Processing Unit IP architecture.

1.5 Multimedia Features

This release supports the media formats and the codecs listed in Table 3.

Table 3: Supported Media Format and Codecs

Containers	Extensions	Audio/Video Combinations		Playback Engine	GST Demuxer Plugins	Status
		Audio	Video			
ASF	.asf	WMA	MPEG-4	GStreamer	asfdemux/gst-plugins-ugly	Ready
	.wmv	WMA	WMV			
	.wma	WMA	No video			
AVI	.avi	MP3	H.264	GStreamer	avidemux/gst-plugins-good	Ready
		MP3	MPEG-4			
		MP3	H.263			
MOV	.mov	AAC	H.264	GStreamer	qtdemux/gst-plugins-good	Ready
		AAC	MPEG-4			
MP4	.mp4	AAC	H.264	Stagefright	--	Ready
MPEG-2 PS	.mpg	MP3	MPEG-2	GStreamer	flupsdemux/ gst-fluendo-mpegdemux	Ready
3GPP	.3gp/.3gpp	AAC	MPEG-4	Stagefright	--	Ready
MKV	.mkv	MP3	H.264	Stagefright	--	Ready
		AAC	H.264			
AAC	.aac	AAC	No video	GStreamer	no demuxer	Ready
MP3	.mp3	MP3	No video	Stagefright	--	Ready
WebM	.webm	Ogg	VP8	Stagefright	--	Ready
TS	.ts	AAC	H.264	Stagefright	--	Ready
AMR	.amr	AMR-NB/WB	No video	Stagefright	--	Ready

1.6 Board Support Package Features

Board support package features for the ARMADA 610 Tablet Reference Platform for Android 2.3 are as follows.

- U-Boot
  - NAND, non-trusted boot
  - USB Ethernet download
  - zImage format support
  - Support for burning Yet Another Flash File System (YAFFS) image
  - Support for burning an image into eMMC
- Linux Kernel 2.6.35
  - L1 cache
  - L2 cache
  - Interrupt controller
  - Peripheral DMA (PDMA) controller
  - Memory controller
  - Real-Time Clock (RTC)
  - Operating System Timer (OST)
  - Intel® Wireless MMX™[1] technology
  - General purpose Input Output (GPIO) interrupt request (IRQ)
  - Clock management
  - Single level cell (SLC) NAND flash memory
  - OneNAND flash memory
  - Journaling Flash File System, version 2 (JFFS2) support
  - Unsorted Block Image File System (UBIFS) support
  - Yet Another Flash File System (YAFFS)
  - MultiMediaCard (MMC3.2 and MMC4.0)
  - Secure Digital (SD)/SDIO (SD1.1 and SD2.0)
  - UART
  - HDMI Audio
  - SSPA drivers
  - ALSA framework
  - Keypad
  - I2C - Normal I2C (see, I2C Stability, on page 12)
  - DSI LCD panel (base frame, overlay)
  - HDMI LCD TV path
  - Audio DMA (ADMA) controller
  - Marvell Wireless Memory Management technology
  - Performance Monitor Unit (PMU) – used by the Linux Oprofile tool
  - USB client
  - Maxim MAX8925 Power Management Integrated Circuit (PMIC) and MAX8649 regulator
  - Battery driver
  - USB charger

1. Intel and MMX and related marks are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

- USB HOST and USB On-the-Go (OTG)
- Video DMA (VDMA) controller
- WM8994 codec
- Dynamic Voltage and Frequency Management (DVFM)
- Multi touch
- On key and Reset
- Wi-Fi, Bluetooth on SD8787
- CM3623 Light and Proximity sensor
- HDMI EDID
- Marvell Vmeta™ technology, a multiple format high-definition video codec supporting multi instances
- ARMv7 mode
- Audio record
- Power off command
- OmniVision® OV5642 5-megapixel SoC sensor support
- LCD additional mode
- Capacity keypad
- TPK800 touch
- AUO panel
- FM audio
- CyWee motion sensor
- High-bandwidth Digital Content Protection (HDCP)
- Extended File System, version 4 (ext4) support

I2C Stability

The power on sequence can impact the I2C stability.

On the Marvell ARMADA 610 Tablet Reference Platform, use the following steps to power on.

1. Unplug the USB cable and the power cable to make sure power is off to the board.
2. Insert the power cable.
3. Press the on-key for about 2 seconds to turn on power to the board.
4. Insert the USB cable.

Following these steps ensures that the board is powered by the DC power, not by the USB cable VBUS.

1.7 Release Package Contents

The following tables list and describe the release package for the ARMADA 610 Tablet Reference Platform for Android™ 2.3, Linux kernel 2.6.35.

Table 4: Prebuilt Binary Files

Files	Description
ARMADA610_ANDROID_GINGERBREAD_PREBUILT_BIN.zip	Prebuilt bin binaries
• primary_gpt_8g	Primary GUID partition table (for 8GB eMMC)
• secondary_gpt_8g	Secondary GUID partition table (for 8GB eMMC)
• system.img	Android system image
• zImage.android	Kernel image
• ramdisk.img	RAM disk image
• u-boot.bin	U-Boot
• nontrusted/ntim_mmp2_nand_bbu_ddr.bin	NTIM header
• nontrusted/ntim_mmp2_nand_ddr3_elipda_lg.txt	NTIM description file
• nontrusted/MMP2_NTLOADER_3_2_19.bin	NTIM loader
• trusted/tim_mmp2_nand_bbu_ddr.bin	TIM header
• trusted/tim_mmp2_ddr3_elipda_lg.txt	TIM description file
• trusted/MMP2_ntloader_3_2_19.bin	TIM loader
• trusted/partition.bin	Partition image for trusted boot
• trusted/partition.txt	Partition description file for trusted boot
• nontrusted/partition.bin	Partition image for non-trusted boot
• nontrusted/partition.txt	Partition description file for non-trusted boot

Note

The WTM firmware image must be downloaded separately from the Marvell Extranet at *My Products/Cellular & Handheld Solutions/Applications Processors/ARMADA 610 (MMP2) Software/WTM/Version 2.1.6*. Contact your Marvell representative if you have issues about the download.

Table 5: Source Files

Files	Description
ARMADA610_ANDROID_GINGERBREAD_SRC	Source code tarball (patch based source code)
• setup_android.sh	Script help to set up the Android code base from the xxx_src.tgz and xxx_patches.tgz
• android_patches.tgz	Marvell patches to the Android Projects
• android_src.tgz	Source code for projects added by Marvell
• marvell_manifest.xml	Manifest xml file to download the Android source code from Google as a base
• kernel_patches.tgz	Marvell patches to kernel_src.tgz
• kernel_src.tgz	Kernel base source code
• uboot_src.tgz	U-Boot base source code
• uboot_patches.tgz	Marvell patches to uboot_src.tgz
• obm_src.tgz	OEM Boot Module (OBM) source code

Caution

The non-trusted image module (NTIM), the trusted image module (TIM), and BootLoader (OEM boot module) files provided are designed and customized for use with the associated Marvell hardware platform. Use these files as a reference. You MUST create the NTIM, TIM, and BootLoader with the correct parameters for your design.

Failure to correctly implement the NTIM, TIM, or BootLoader may result in a boot failure or cause an unreliable operation of your device.

For information and assistance in correctly setting up your NTIM, TIM, and BootLoader, see the Marvell Boot ROM or Marvell Wireless Trusted Tool Package documentation or contact your Marvell Applications Engineer or Field Applications Engineer.

Note

For detailed information about the WTPTP release package, see the *Marvell® Wireless Trusted Platform Tool Package for Application Processors Software Release Notes* (MV-S301673-00).

2 Installation

This section provides procedures for

- Identifying an ARMv6 or ARMv7 mode boot
- Programming the binaries onto flash memory with the Marvell eXtreme Debugger
- Downloading Android onto flash memory
- Setting up the Android working directory
- Building Android
- Using the optimized OpenSSL for Marvell platforms
- Protecting the HDCP key

Note

See Section 1.7, Release Package Contents, on page 13 for a description of the release package contents.

2.1 Use the Prebuilt Binaries

The following procedures provide information for programming binaries onto flash memory using the Marvell eXtreme Debugger.

2.1.1 Identifying an ARMv6 or ARMv7 Mode Boot

The non-trusted image module (NTIM) or the trusted image module (TIM) is updated to switch the processor core from ARMv6 to ARMv7 mode. If the new NTIM or TIM image is burned into flash correctly, the boot ROM switches the core from ARMv6 to ARMv7 mode during the boot. Without a new NTIM, the processor still boots in the ARMv6 mode.

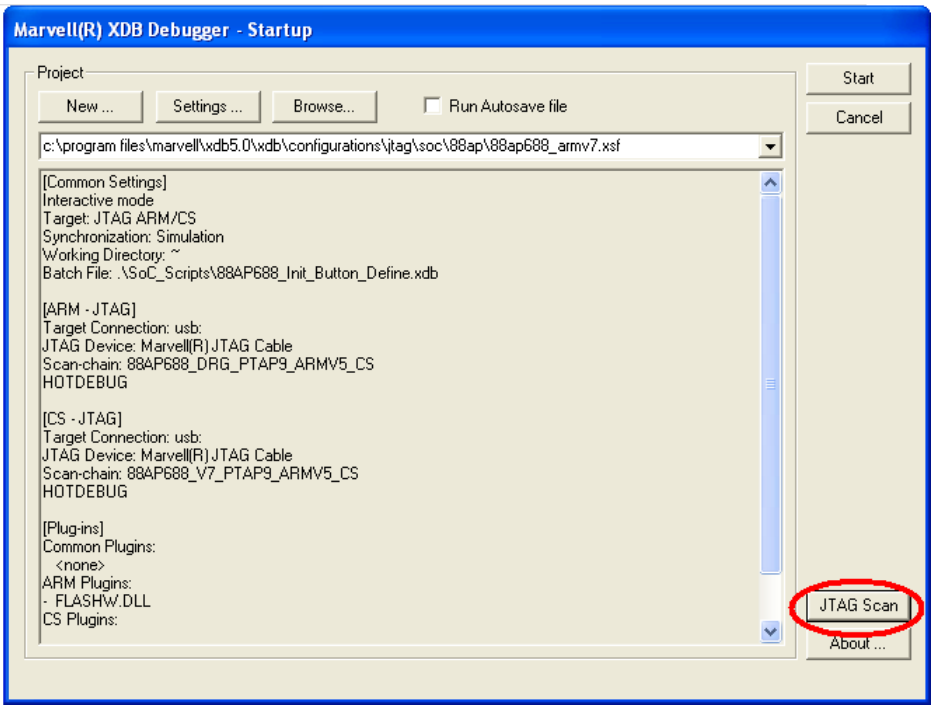
To identify whether the processor is in ARMv6 or ARMv7 mode, use the steps that follow.

Note

You need the Marvell eXtreme Debugger, version 5.2 Beta 1 or later. Download the Marvell eXtreme Debugger from the Marvell Extranet or contact your Marvell representative.

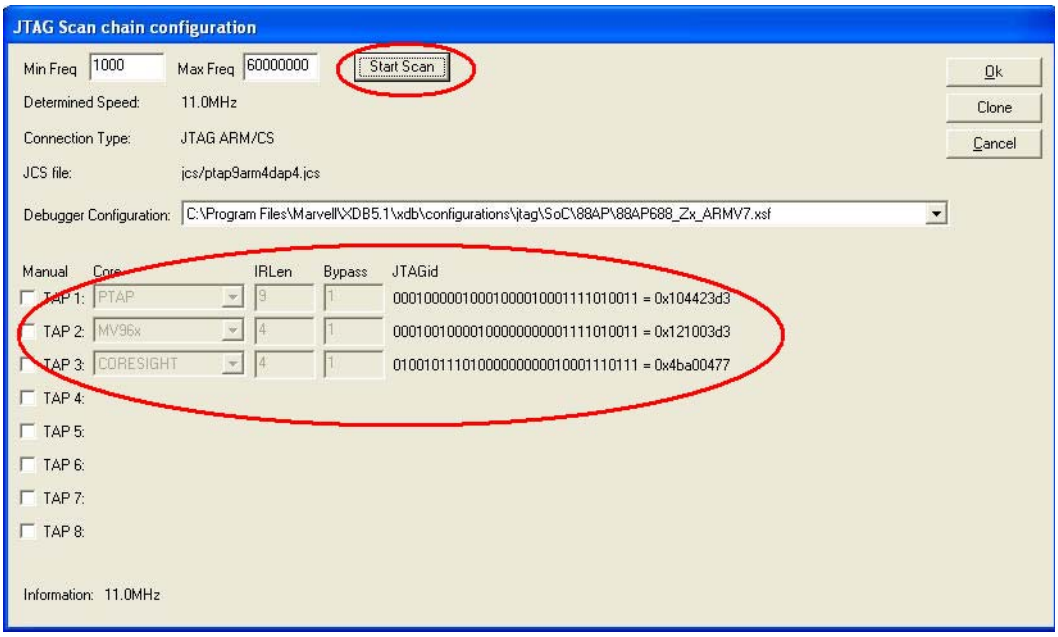
1. With the Marvell eXtreme Debugger (XDB) running, click the **JTAG Scan** button on the Startup screen.

Figure 1: XDB Debugger JTAG Scan Button



2. Click the **Start Scan** button.
   Figure 2 shows a scan chain that indicates the core mode is in ARMv7 mode.
   - Three Test Access Ports (TAPs) indicate the core is in ARMv7 mode.
   - Four TAPs indicate the core is in ARMv6 mode

Figure 2: Start Scan Button and JTAG Scan Chain



2.1.2 Burning the Trusted WTM Firmware, OEM Boot Module, and U-Boot Binaries to eMMC Using JTAG

Perform the steps that follow to burn the trusted WTM firmware, OEM boot module (OBM), and U-Boot binaries to eMMC flash memory using JTAG.

Note You need the Marvell eXtreme Debugger, version 5.2 Beta 1 or later. Download the Marvell eXtreme Debugger from the Marvell Extranet or contact your Marvell representative.

1. Use the Marvell eXtreme Debugger (XDB), version 5.2 Beta 1 and the appropriate configuration file for the steps that follow:
   - If the processor boots in the ARMv6 mode, use the `pxa688 _a0_armv6.xsf` file.
   - If the processor boots in the ARMv7 mode, use the `pxa688 _a0_armv7.xsf` file.

   Leave all settings at their default settings.

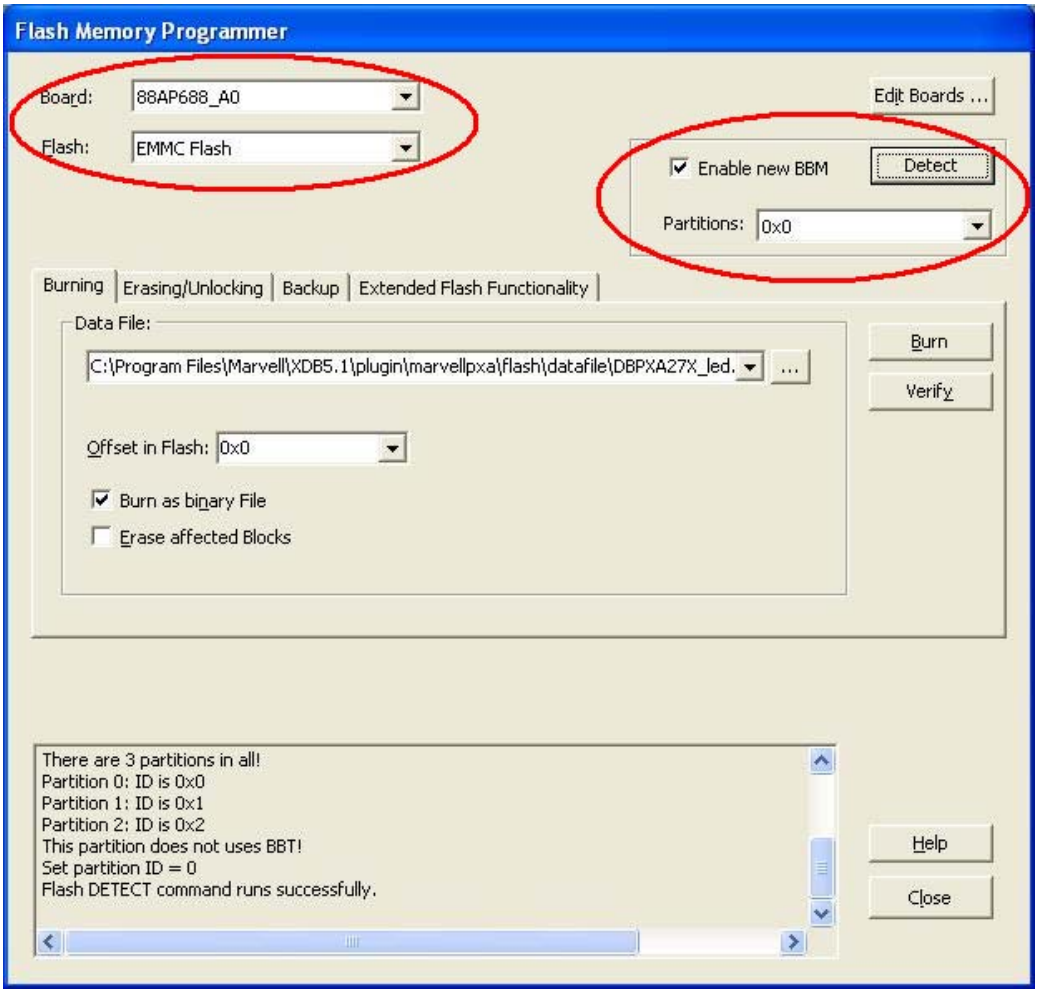
Note To identify in which mode your processor boots, see Section 2.1.1, Identifying an ARMv6 or ARMv7 Mode Boot, on page 15.

2. With XDB running, select **Flash** on the toolbar menu.
3. Click **Burn Flash** on the drop-down menu.
4. In the **Board** field, select **88AP688_A0**, and in the **Flash** field, select **EMMC Flash** (see Figure 3 on page 19).

Note If you are using the 88AP688 Z0/Z1 stepping processor, select **88AP688_ZX** in the **Board** field instead.

5. Check the **Enable new BBM** check box and click the **Detect** button to detect the eMMC partition (see Figure 3).

Figure 3: eMMC Flash Detect eMMC Partition



6. Select partition **0x1**. Burn the following binary files to the addresses as follows:

```
tim_mmp2_nand_bbu_ddr.bin          --> 0x0
partition.bin                      --> 0x600
Wtm_rel_mmp2.bin                   --> 0x1000
MMP2_TLOADER_3_2_19.bin            --> 0x32000
u-boot.bin                         --> 0x52000
```

The `Wtm_rel_mmp2.bin` image is a loadable WTM kernel firmware binary image that is executed by the ARMADA 610 on-chip secure processor. This image provides the cryptographic services for both Federal Information Processing Standard (FIPS) and non-FIPS mode operations.

To acquire the image, download the `WTM_Firmware_ARMADA610_2.1.6.zip` file from the Marvell Extranet at *My Products/Cellular & Handheld Solutions/Applications Processors/ ARMADA 610 (MMP2)/Software/WTM/Version 2.1.6*.

Once downloaded, extract both `wtm_rel_mmp2_virtualOTP_2.1.6.bin` and `wtm_rel_mmp2_realOTP_2.1.6.bin` images from the zip file. Both binary images support the same set of WTM primitive functions with the same API definition. However, the `wtm_rel_mmp2_virtualOTP_2.1.6.bin` binary image performs the device RKEK/EC521-DK provision with primitives using the buffers within the secure SRAM to emulate the provisioning over the FUSE/OTP macro. On the other hand, the `wtm_rel_mmp2_realOTP_2.1.6.bin` image performs the device key provision with primitives directly operating over the FUSE/OTP macro. With real OTP operation, the performed platform provision becomes permanent.

It is recommended to use the virtual OTP version of the WTM kernel binary image for platform software development. To do this, change the binary image file name `wtm_rel_mmp2_virtualOTP_2.1.6.bin` to `Wtm_rel_mmp2.bin`.

It is recommended to use the real OTP version of the WTM kernel binary image for the device that is ready to be deployed as a product. To do this, change the binary image file name `wtm_rel_mmp2_realOTP_2.1.6.bin` to `Wtm_rel_mmp2.bin`.

2.1.3 Burning the Non-trusted WTM Firmware, OBM, and U-Boot Binaries to eMMC Using JTAG

Perform the steps that follow to burn the non-trusted WTM firmware, OBM and U-Boot binaries to eMMC flash memory using JTAG:

Note You need the Marvell eXtreme Debugger, version 5.2 Beta 1 or later. Download the Marvell eXtreme Debugger from the Marvell Extranet or contact your Marvell representative.

1. Use the Marvell eXtreme Debugger (XDB), version 5.2 Beta 1 or later and the appropriate configuration file for the steps that follow:
   - If the processor boots in the ARMv6 mode, use the `pxa688 _a0_armv6.xsf` file.
   - If the processor boots in the ARMv7 mode, use the `pxa688 _a0_armv7.xsf` file.

   Leave all settings at their default settings.

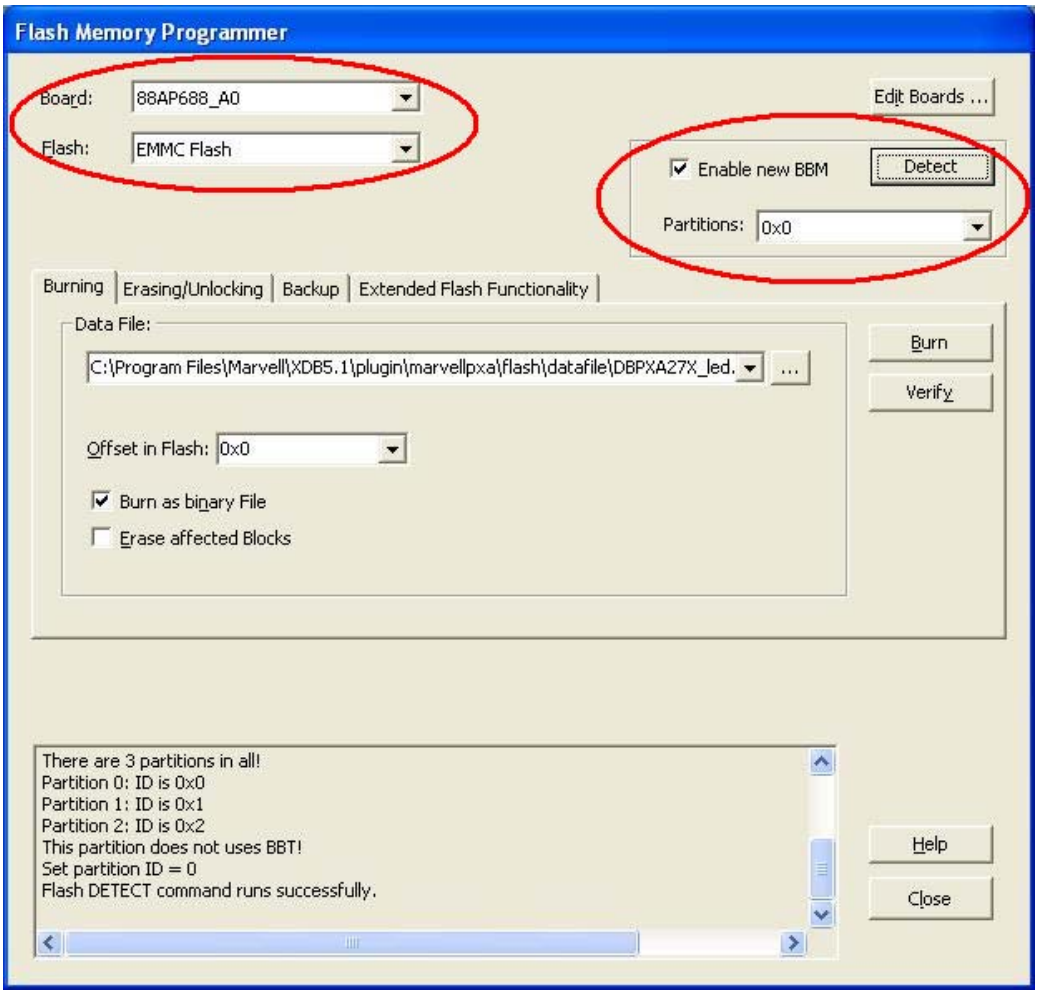
Note To identify in which mode that your processor boots, see Section 2.1.1, Identifying an ARMv6 or ARMv7 Mode Boot, on page 15.

2. Burn OBM and U-Boot into eMMC flash memory using XDB:
   a) In XDB, select **Flash** on the toolbar menu. Click **Burn Flash** on the drop-down menu.
   b) In the **Board** field, select **88AP688_A0**, and in the **Flash** field, select **EMMC Flash** (see Figure 4 on page 22).

Note If you are using the 88AP688 Z0/Z1 stepping processor, select **88AP688_ZX** in the **Board** field instead.

   c) Check the **Enable new BBM** check box and click the **Detect** button to detect the eMMC partition (see Figure 4).

Figure 4: eMMC Flash Detect eMMC Partition



d) Select partition **0x1**. Burn the following binary files to the addresses as follows:

```
ntim_mmp2_nand_bbu_ddr.bin          --> 0x0
partition.bin                       --> 0x600
Wtm_rel_mmp2.bin                    --> 0x1000
MMP2_NTLOADER_3_2_19.bin            --> 0x32000
u-boot.bin                          --> 0x52000
```

The `Wtm_rel_mmp2.bin` image is a loadable WTM kernel firmware binary image that is executed by the ARMADA 610 on-chip secure processor. This image provides the cryptographic services for both Federal Information Processing Standard (FIPS) and non-FIPS mode operations.

To acquire the image, download the `WTM_Firmware_ARMADA610_2.1.6.zip` file from the Marvell Extranet at *My Products/Cellular & Handheld Solutions/Applications Processors/ ARMADA 610 (MMP2)/Software/WTM/Version 2.1.6*.

Once downloaded, extract both `wtm_rel_mmp2_virtualOTP_2.1.6.bin` and `wtm_rel_mmp2_realOTP_2.1.6.bin` images from the zip file. Both binary images support the same set of WTM primitive functions with the same API definition. However, the `wtm_rel_mmp2_virtualOTP_2.1.6.bin` binary image performs the device RKEK/EC521-DK provision with primitives using the buffers within the secure SRAM to emulate the provisioning over the FUSE/OTP macro. On the other hand, the `wtm_rel_mmp2_realOTP_2.1.6.bin` image performs the device key provision with primitives directly operating over the FUSE/OTP macro. With real OTP operation, the performed platform provision becomes permanent.

It is recommended to use the virtual OTP version of the WTM kernel binary image for platform software development. To do this, change the binary image file name `wtm_rel_mmp2_virtualOTP_2.1.6.bin` to `Wtm_rel_mmp2.bin`.

It is recommended to use the real OTP version of the WTM kernel binary image for the device that is ready to be deployed as a product. To do this, change the binary image file name `wtm_rel_mmp2_realOTP_2.1.6.bin` to `Wtm_rel_mmp2.bin`.

2.1.4 Burning Android on eMMC Using U-Boot

Use the U-Boot commands that follow to tftp `zImage.android`, `primary_gpt_8g`, `secondary_gpt_8g`, `ramdisk.img`, and `system.img` to eMMC flash memory.

Note

The following procedures require the connection of the USB cable for USB Ethernet between the host Linux PC and the ARMADA 610 Tablet Reference Platform.

After the first TFTP command is issued, use the ifconfig utility to set the host side USB0 connection to `192.168.1.100`. The Network File System (NFS) server is fixed as `192.168.1.100:/nfs/android`. If you want to change it, modify `vendor/marvell/brownstone/rootdir/rdinit` and build the kernel again.

You do not see the USB0 Ethernet interface on the Linux host until AFTER the first TFTP “t” command is issued from U-boot.

1. By default, the initial partition is set to 0. Alternately, you can run the following instructions to make sure.

```
MMP2       --> mmc sw_part 0
```

2. Burn GPT.

```
MMP2       --> t primary_gpt_8g
MMP2       --> mmc write 0 0x22 0x1100000
MMP2       --> t secondary_gpt_8g
MMP2       --> mmc write 0xecafdf 0x21 0x1100000
```

3. Burn the kernel

```
MMP2       --> t zImage.android
MMP2       --> mmc write 0x4c00 0x2000 0x1100000
```

4. Burn the ramdisk image.

```
MMP2       --> t ramdisk.img
MMP2       --> mmc write 0x8C00 0x800 0x1100000
```

5. Burn the system image.

```
MMP2       --> t system.img
MMP2       --> mmc write 0xCC00 0x4b000 0x1100000
```

6. Clean up the data.

```
MMP2       --> mw.b 0x1100000 0 0x80000
MMP2       --> mmc write 0x57c00 0x400 0x1100000
```

7. Recover the images.

```
MMP2        --> t zImage_recovery.android
MMP2        --> mmc write 0xa2c00 0x2000 0x1100000
MMP2        --> t ramdisk_recovery.img
MMP2        --> mmc write 0xa6c00 0x800 0x1100000
```

Note

When you are done, power off the ARMADA 610 Tablet Reference Platform and power it on again to boot from eMMC.

Rebooting by pressing the reset button does not work.

2.2 Use the Patch-based Source Code

The source code package includes the code for the kernel, U-Boot, Android, and everything needed to boot Android on an ARMADA 610 Tablet Reference Platform with the ARMADA 610 processor with A0/A1/A2 stepping and 1GB of DDR3 memory.

The Android code is provided as a group of patches based on a certain version of Android source code. A manifest file is provided to download that version of Android code from the Android Open Source Project (AOSP) at http://source.android.com.

The kernel and U-Boot are provided as a tar ball of base code and a tar ball of patches Marvell made on it.

2.2.1 Setting Up the Android Working Directory

Use the steps that follow to set up the code base.

Note

Check the version of your Git. You can do this by typing `git version`. If the Git version is 1.6.x.x. You can go ahead with it.

If the Git version is 1.7.1.x or later, open the `~/.gitconfig` file and add the following section:

```
[am]
    keepcr=true
```

If the Git version is 1.7.0.x, upgrade your Git to a version later than 1.7.1.x. You can download the package from http://git-scm.com/download.

1. Go to http://source.android.com to download the "repo" tool and set up the build environment for Android.
2. Create the Android working directory and download the initial code base.

```
$ mkdir <android_working_dir>
$ cd <android_working_dir>
$ repo init -u git://android.git.kernel.org/platform/manifest -b master
$ repo sync
```

3. Switch the code base specified by `marvell_manifest.xml`.

```
$ cp <installed_source_dir>/marvell_manifest.xml .repo/manifests/
$ repo init -m marvell_manifest.xml
$ repo sync
```

4. Apply the Marvell patches:

```
$ cd <installed_source_dir>
$ ./setup_android.sh <android_working_dir>
```

2.2.2 Building the Source Code

Use the steps that follow to build Android.

1. Build the kernel and modules:

```
$ cd <android_working_dir>
$ cd kernel
$ make all
```

Note The location of the zImage is at `kernel/out/`and the location of the modules is at `kernel/out/modules`.

2. Build U-Boot and OBM:

   Before building U-Boot and OBM, extract `wtm_rel_mmp2_virtualOTP_2.1.6.bin` from `WTM_Firmware_ARMADA610_2.1.6.zip` which you can get from the Marvell Extranet. Rename it as `Wtm_rel_mmp2.bin` and copy it to `boot/obm/binaries`.

Note The WTM firmware image must be downloaded separately from the Marvell Extranet at *My Products/Cellular & Handheld Solutions/Applications Processors/ARMADA 610 (MMP2) Software/WTM/Version 2.1.6*. Contact your Marvell representative if you have issues about the download.

   Issue the following commands:

```
$ cd <android_working_dir>
$ cd boot
$ make all
```

Note The `u-boot.bin` and `OBM` files are at `boot/out/nontrusted`, while the unified WTM files are at `boot/out/`.

3. Build android:

```
$ cd <android_working_dir>
$ . build/envsetup.sh
$ chooseproduct brownstone
$ export ANDROID_PREBUILT_MODULES=kernel/out/modules/
$ make -j4
```

Note

The location of the Android GPTs, `ramdisk.img`, `system.img`, and `update_droid.zip` files are at `out/target/product/brownstone`.

2.2.3 Using the Optimized OpenSSL for Marvell Platforms

An OpenSSL patch is included in this platform release. This patch fixes a SHA384/512 bug in Android OpenSSL, and optimizes the Android OpenSSL cryptographic library for Marvell platforms. By default, the SHA384/512 bug fix is enabled, the optimization is disabled.

To enable the optimization, this patch requires the Marvell Wireless Trusted Platform Service Package (WTPSP) in this platform release. See the WTPSP release notes for information about how to enable it. Contact your Marvell representative if you have any issues with this package.

When the optimization is enabled, the following OpenSSL cryptographic schemes are optimized: SHA1/224/256, MD5 message digest, AES (CBC mode), RC4 and DES (CBC, CBC3 mode).

Note

Whether the optimization is enabled or not enabled, the Android OpenSSL cryptographic API stays unchanged. Thus, applications using the OpenSSL cryptographic library do not need to be modified. However, an application rebuild is required when the optimization is enabled.

To enable Marvell optimization for OpenSSL:

1. Put the Marvell WTPSP middleware library and header file into `<android_working_dir>/external/openssl/crypto/` and rename it as `libwtpsp.a`.
2. Add the following into `external/openssl/include/openssl/opensslconf.h`.

```
#ifdef _ARM_
#ifndef OPENSSL_MRVL
#define OPENSSL_MRVL /* enable marvell crypto support */
#endif
#endif
```

3. Add the following into `vendor/marvell/brownstone/BoardConfig.mk`.

```
USE_MARVELL_CRYPTO := true
```

4. Build the Android system image.

2.3 Protect the HDCP Key

The ARMADA 610 platform supports the high-bandwidth digital content protection (HDCP) protocol that is communicated across the display port, digital visual interface (DVI), high-definition multimedia interface (HDMI), gigabit video interface (GVIF), or unified display interface (UDI) connection.

To protect the device unique HDCP key set (40 keys with each 56 bits) used for streaming multimedia data ciphering, as well as the key select vector (40-bit KSV) for the HDCP authorization, HDMI/HDCP service in the Android package is capable of wrapping the HDCP key set during device HDCP provisioning through the WTPSP and WTM, and loading the HDCP key set into the platform HDCP key register during the platform booting sequence through the WTPSP and WTM.

Note

Marvell does not provide any HDCP key. To enable HDCP, get the HDCP key from Digital Content Protection, LLC: http://www.digital-cp.com/

To protect the HDCP key on the ARMADA 610 platform, use the steps that follow.

1. Put the plaintext HDCP key at `/data/HDCP/p_key.img`.
2. At the very first time HDMI service is launched in the Device Manufacturing (DM) life cycle, the plaintext HDCP key is wrapped to a cipher key. The cipher key is saved at `/data/HDCP/c_key.img`. The plaintext key is deleted.
3. Once the HDCP key is wrapped, HDMI service loads the cipher key into the platform HDCP key register.

Note

The format of the plaintext HDCP key, `p_key.img`, should be 5 bytes of the KSV + 3 bytes of 0x00 + 280 bytes of private keys.

Note

HDCP key wrapping can be done multiple times if desired, but must be done in the DM life cycle. Once the Device Deployment (DD) life cycle is entered, only the cipher HDCP key can be loaded into the platform HDCP key register.

3 Recovery and Updates

This section provides procedures for

- Using the recovery and update features
- Checking device requirements
- Checking the kernel and U-Boot requirements
- Customizing the update

3.1 Recovery and Update Features

The following sections provide information about the following features:

- Burning recovery images
- Entering recovery mode
- Performing a factory reset
- Using the update package
- Using fastboot

3.1.1 Burning Recovery Images

To enable the recovery and update features, two additional images are required:

- `zImage_recovery.android`
- `ramdisk_recovery.img`

Burn these images to eMMC flash memory as follows.

```
MMP2 --> t zImage_recovery.android
MMP2 --> mmc write 0xa2c00 0x2000 0x1100000
MMP2 --> t ramdisk_recovery.img
MMP2 --> mmc write 0xa6c00 0x800 0x1100000
```

3.1.2 Entering Recovery Mode

On the Marvell ARMADA 610 Tablet Reference Platform, the system enters Recovery mode if the user powers on the device and presses the Home button. An icon is displayed when the system enters Recovery mode.

Figure 5: Recovery Mode Icon

Pressing the Home button switches the screen between the Recovery menu and the background icon.

The Recovery menu contains four options:

- reboot system now
- apply update from /sdcard
- wipe data/factory reset
- wipe cache partition

Use the Menu and Back buttons to move the highlight to the desired option. Use the Search button to select the option.

When the “apply update from /sdcard” option is selected and the system is installing the update, the background is set to the installing icon (see Figure 6).

Figure 6: Installing Icon

3.1.3 Performing a Factory Reset

The factory reset is a user interface interaction which erases data and the cache partition. If "Settings => Privacy => Factory data reset" is selected, the system reboots in Recovery mode to erase data and cache.

3.1.4 Using the Update Package

To update using the SD card, put the update package on the SD card and reboot the device by pressing the appropriate button to enter Recovery mode (see Section 3.1.2, Entering Recovery Mode, on page 29). Then, select the menu option "apply update from /sdcard," select the update package, and start the update.

To update in Over-the-Air (OTA) mode, implement an Android service to check if a new update package is ready in the OTA server, download the update package to the device, and reboot the device in Recovery mode to process the update.

3.1.5 Using Fastboot

Fastboot is a protocol used to update the flash filesystem in Android devices from a host over USB. The system enters the Fastboot mode if the user powers on the device and presses the Menu button. A partition table is displayed in the host console when the system enters the Fastboot mode.

To make use of fastboot, you need the fastboot program compiled for your host computer (the location should be `out/host/linux-x86/bin/fastboot`).

Here are the commands you can run on your host after fastboot has been started on a device connected via USB. You can get command hints from the fastboot binary's "--help" command:

Usage: `fastboot [ <option> ] <command>`

Commands:

`flash <partition> [ <filename> ]`	Writes a file to a flash partition
`erase <partition>`	Erases a flash partition
`reboot`	Reboots the device normally

Options:

`-w`	Erases userdata and cache
`-s <serial number>`	Specifies the device serial number
`-p <product>`	Specifies the product name

You can use fastboot to burn all images to eMMC flash memory as follows. If the target image is in current path, you can omit the filename option.

```
# fastboot flash kernel [<image path>/zImage.android]
# fastboot flash ramdisk [<image path>/ramdisk.img]
# fastboot flash system [<image path>/system.img]
# fastboot flash reckernel [<image path>/zImage_recovery.android]
# fastboot flash recovery [<image path>/ramdisk_recovery.img]
```

You can put all images in the current path and use `flashall` to burn all images with one command:

```
# fastboot flashall
```

3.2 Device Requirements

3.2.1 Key Layout

The ARMADA 610 Tablet Reference Platform boots normally when the On/Off power button is pressed. The device enters a Recovery mode if the user presses the appropriate button.

Several other keys (buttons) are required to operate the menu in the Recovery mode, such as the Home and Back buttons for moving the highlight to an option, the Search button for selecting an option, and so on. See Section 3.1.2, Entering Recovery Mode, on page 29 for a description of these buttons.

For the location and description of the ARMADA 610 Tablet Reference Platform buttons, see the *ARMADA 610 Tablet Reference Platform User's Guide*, MV-L100771-100.

3.2.2 eMMC Partition

To enable the recovery feature, four more partitions must be added:

- kernel recovery partition
- RAM disk recovery partition
- misc partition
- cache partition

The misc partition is used for interaction between recovery and the BootLoader. The cache partition is used for interaction between recovery and the main system.

Use the GUID Partition Table (GPT) partition method instead of Master Boot Record (MBR) to support more partitions. Use the "parted" command to generate the GPT table; include the primary and secondary GPT header.

1. Make a partition image with the same size as the on-board flash memory. For example, generate an 8GB image file, `gpt.img`:

   ```
   # dd if=/dev/zero of=gpt.img bs=512 count=15511551  // 15511551 blocks in 8GB flash, 512 bytes per block
   ```

2. Format and partition the image file:

   a) Enter the partition program.

   ```
   # parted gpt.img
   ```

   b) Format the image file as GPT.

   ```
   # (parted) mklabel gpt
   ```

   c) Change the unit to byte.

   ```
   # (parted) unit b
   ```

   d) Add a new partition "ramdisk".

   ```
   # (parted) mkpart
   # Partition name? [ ]? ramdisk
   # File system type? [ext2]?
   # Start? 18350080         // 0x8c00 * 512 = 18350080
   # End? 26738687           // 0xcc00 * 512 - 1 = 26738687
   ```

   e) You can also rename the partition.

   ```
   # (parted) name 1         // the number is the index number of the partition
   # Partition name? [ramdisk] ramdisk
   ```

f) Add more partitions; follow the previous steps.

Note

1. At any time, you can use the "print" command to check the partition information.
   `# (parted) print`
2. The order you create new partitions is the order listed in the GPT table. It is the order recognized in uboot/kernel.
   If you create partitions in the following sequence: ramdisk -> android_system -> android_data -> … Then, partition "ramdisk" is mmcblk0p1, "android_system" is mmcblk0p2, "android_data" is mmcblk0p3, … .
   If you create partitions in this sequence: kernel -> ramdisk -> android_system -> … Then, partition "kernel" is mmcblk0p1, "ramdisk" is mmcblk0p2, "android_system" is mmcblk0p3, … .
3. At any time, you can type "?" for more details for help.
   `# (parted) ?`

g) Quit the partition program.

```
# (parted) quit
```

4. Get the GPT table from the image file:
   # dd if=gpt.img of=primary_gpt bs=512 count=34
   # dd if=gpt.img of=secondary_gpt bs=512 count=33 skip=15511519

On Marvell ARMADA 610 Tablet Reference Platform, the partition table is as follows.

Partition	Start Block Address	End Block Address	Block Size	Byte Size	Device Name
kernel	0x4c00	0x8c00	0x4000	8MB	mmcblk0p8
ramdisk	0x8c00	0xcc00	0x4000	8MB	mmcblk0p1
android system	0xcc00	0x57c00	0x4b000	150 MB	mmcblk0p2
android data	0x57c00	0xa2c00	0x4b000	150 MB	mmcblk0p3
kernel recovery	0xa2c00	0xa6c00	0x4000	8MB	mmcblk0p9
ramdisk recovery	0xa6c00	0xaac00	0x4000	8MB	mmcblk0p5
misc	0xaac00	0xab000	0x400	512 KB	mmcblk0p6
cache	0xab000	0xf6000	0x4b000	150 MB	mmcblk0p7
mass storage	0xf6000	0xecafdf	0xdd4fdf	7082 MB	mmcblk0p4

3.3 Kernel and U-Boot Requirements

The following sections list kernel and U-Boot requirements.

3.3.1 Kernel Requirements

The `reboot()` system call with the "recovery" parameter must set a register bit to inform the BootLoader to enter Recovery mode and to reset the bit after entering Recovery mode.

3.3.2 U-Boot Requirements

U-Boot must perform the following procedures:

- Detect the appropriate key and enter the Recovery mode if the Home button key is pressed.
- Check the recovery indication register bit and enter the Recovery mode if the bit is set, then reset the bit.
- Read the misc partition, if the buffer is equal to "boot-recovery", enter the Recovery mode; if the buffer is equal to "update-firmware," get the BootLoader image from misc and reflash itself.
- Support fastboot protocol. Detailed information is at the following location:
  `bootable/bootloader/legacy/fastboot_protocol.txt`

3.4 Update Customization

The following sections describe how to customize generating the update package, signature, and certification.

3.4.1 Generating the Update Package

The update package is a ZIP file containing an update binary, an update script, some certification information, and all the files needed for the update. The directory structure is as follows.

```
|-- META-INF
|   |-- CERT.RSA
|   |-- CERT.SF
|   |-- MANIFEST.MF
|   |-- com
|       |-- google
|           |-- android
|               |-- update-binary
|               |-- updater-script
|-- system/…
|-- <other folders>/…
|-- zImage
|-- <other images>
```

The update packages can be generated automatically in the Android system build. In the system, the update binary and all the files needed for the update are copied to a specific folder. In addition, the update script is generated by several Python tools.

If you want to customize an update package, for example, just update one or several `.so` files or `apk` files, generate the package manually as follows.

1. Make an update folder and copy all folders and files into it:
   ```
   # mkdir <update folder>
   ```
   ```
   # cp -p -r <folds and files need to update> <update folder>
   ```
2. Make a meta info folder, copy the updater binary to it, and create an update script:
   ```
   # mkdir -p META-INF/com/google/android
   ```
3. Compress the update folder:
   ```
   # zip -qry ../update.zip .
   ```

3.4.2 Generating the Signature and Certification

By default, the update process uses the Android test key pair to sign the update package, `testke.x509.pem` and `testkey.pk8` in `build/target/product/security/`. The signature using the private key is done in the build process and the public key is dumped to the recovery RAM disk at the same time. The recovery process verifies the update package before extracting and upgrading the package.

To use a customized key, follow these steps:

1. Generate the private key:
   ```
   # openssl genrsa -3 -out customer_key.pem 2048
   ```
2. Generate the certification:
   ```
   # openssl req -new -x509 -key customer_key.pem  -out customer_key.x509.pem
   -days 10000
   ```
3. Turn the private key to PKCS #8 standard:
   ```
   # openssl pkcs8 -in customer_key.pem -topk8 -outform DER -out customer_key.pk8
   -nocrypt
   ```
4. Sign the `update.zip` by `<android>/out/host/linux-x86/framework/signapk.jar`:
   ```
   # java -Xmx512m -jar signapk.jar -w customer_key.x509.pem customer_key.pk8
   update.zip update_signed.zip
   ```
5. Using `<android>/out/host/linux-x86/framework/dumpkey.jar`, dump the public key and build in the recovery ramdisk:
   ```
   # java -jar dumpkey.jar customer_key.x509.pem > keys
   ```

THIS PAGE INTENTIONALLY LEFT BLANK

4 Marvell Code Performance Analyzer

The Marvell Code Performance Analyzer v2.3 is supported in this release.

4.1 What's New

- Call Stack sampling data collection and corresponding data analysis
- Remote data collection via the Android Debug Bridge (ADB) for the Android device

4.2 Features

The following sections provide information about the supported and unsupported features for this release.

4.2.1 Supported Features

- Remote data collection via TCP/IP
- Remote data collection via ADB for the Android device
- Target local data collection in a connectionless environment
- Call Stack sampling data collection and corresponding data analysis
- Hotspot sampling data collection and corresponding data analysis
- Counter monitor data collection and corresponding data analysis.
- Real-time counter monitor and post analysis

4.2.2 Unsupported Features

The software development kit (SDK) for dynamic code is not supported in this release.

4.3 System Requirements

This release supports the Marvell Code Performance Analyzer, version 2.3. Download this version from the Marvell Extranet website at http://www.marvell.com/extranet. If you do not have a Marvell Extranet user ID, click on the “register” link at http://www.marvell.com and follow the instructions therein.

4.4 Installation

Before running the data collector, go to the `/system/bin` folder and run the following command to load the kernel driver:

```
$ ./load_mpdc.sh
```

4.5 Known Issues or Limitations

- It is recommended to first turn off the Marvell Scalable Power Management (MSPM). Otherwise, the Marvell Performance Data Collector (mpdc) may not work normally. Use the following commands to turn off MSPM on your target:

  ```
  echo 0 >/sys/power/mspm/mspm
  ```

- When using the command line, if you get the error message "Fail to communicate with daemon:Success", reload your activity and run it to collect the result again.

- When doing remote data collection on the Android target, if the analyzer exits abnormally, the status of the mpdc_svr may still be connected. If this happens, restart the mpdc_svr on the target.
- If the samples/second value is set too large, the events/sample value is adjusted after a calibration to a smaller value, which makes the interrupt happen more frequently. If this happens, the system becomes busy and the mpdc stops after the expected duration is expired. In addition, most of the samples will be located on the process "mpdc_d". In this case, create your activity with a decreased samples/second value and start it.
- When using the command line, multiple activities cannot be started at the same time.
- If you want to make mpdc_svr listen to another port through using the command "mpdc_svr -p <PortNumber>" on the Android target, a segmentation fault occurs.
- Do not set OS_TIMER as the event in the Event-Based Sampling (EBS).

5 Known Issues

This section describes the Known Issues for the ARMADA 610 Tablet Reference Platform for Android 2.3.

5.1 Fixed Issues

The following issues are fixed in this release.

- [MMP2PV-237] Copying a file about the size of 8MB to a USB drive causes an IO error. (Reading a file from a USB drive is OK.) Use a USB OTG interface instead of a USB High-Speed Inter-Chip (HSIC) interface.
- [MMP2PV-474] When a binder error occurred during the Graphics Abstract Layer 2D (GAL2D) stress test and LCD+SD+WIFI+USB stress testing, the system was unable to restore the Android GUI.
- [MMP2PV-483] An I2C read/write operation via Audio/Light Sensor/Camera with a resume from suspend mode causes the system to hang.
- [MMP2PV-497] A JPEG capture does not work. (There is no data output at the polling call; the YUV format works.)
- [MMP2PV-499] The screen displays abnormally with flickering and some stripes.
- [MMP2PV-507] The system hangs when it resumes from being in suspend mode longer than 4 hours during SD/eMMC read/write operations.
- [MMP2PV-510] Running 15 threads of read/write eMMC and audio/video playback (720P H.264) causes the system to hang.
- [MMP2PV-516] When running the Wi-Fi iPerf stress test longer than 4 hours, a resume form suspend mode causes kernel panic.
- [MMP2PV-537] Bluetooth can still operate in airplane mode.
- [MMP2PV-538] Using the Audio/Video Remote Control Profile (AVRCP), the previous song operation failed.
- [MMP2PV-541] Some video clips cannot be switch to HDMI display.
- [MMP2PV-542] In some streams, audio cannot be switched back from HDMI to the target.
- [MMP2PV-543] The first few frames (about 2 seconds) do not play smoothly after switching some streams to HDMI.
- [MMP2PV-545] The system cannot enter suspend mode after performing a power test.
- [MMP2PV-546] eMMC/SD read/write performance on the Alpha 1 version of the platform with Android 2.3 degrades more than10% compared with the Beta 2.1 RC version of the platform with Android 2.2.
- [MMP2PV-547] The camcorder function does not work.
- [MMP2PV-549] The system cannot enter suspend mode when the on-key is pressed during the video playback.
- [MMP2PV-551] The 720p/1080p UYVY format preview displays with streaks.
- [MMP2PV-554] The IOZONE read/write performance test on eMMC causes the system to hang.
- [MMP2PV-555] The camera preview screen in the camera application sometimes hangs. (The failure rate is 10 percent.)
- [MMP2PV-567] Wi-Fi WPA2 TCP throughput expectations failed.
- [MMP2PV-568] Some audio track tests hang after an "AudioTrack.write" call.

- [MMP2PV-569] Only some channels (about 3 or 4) can be scanned in the ARMADA 610 Tablet Reference Platform.
- [MMP2PV-570] The FM application sometimes does not respond when adjusting the channel.
- [MMP2PV-573] Some video clips require 10 to 15 seconds to switch the video to HDMI display.
- [MMP2PV-574] Some video clips have no audio output during video playback through HDMI display.
- [MMP2PV-575] Some video clips cannot play when HDMI is connected.
- [MMP2PV-578] The console cannot be used in the early suspend state.
- [MMP2PV-585] The Monkey test aborted after running for 45 minutes in the Beta 1 release. (This issue is closed since it is a duplicate of the MMP2PV-799 issue.)
- [MMP2PV-587] Copying files between the WIFI NFS and USB Ethernet NFS stress tests causes the system to hang.
- [MMP2PV-588] The MV_VPPOBJ_Create function returns an error code although this function is executed correctly.

  The error code return is due to the failure of message queue initialization. Remove the message queue initialization part.
- [MMP2PV-590] The 1080p and 720p resolutions work abnormally. There is no video output when resolution is set the first time and the signal is normal; there is video output after setting the resolution the second time.
- [MMP2PV-591] The audio outputs both device and HDMI display under normal mode without switching the video to HDMI.
- [MMP2PV-593] The video playback works abnormally when HDCP is disabled if the enable setting is not previously called.
- [MMP2PV-594] Memory write throughput in this release degrades 10% compared with the Alpha 1 release (Alpha 1: 1479 MB/s --> Beta 1: 1278 MB/s).
- [MMP2PV-595] The LCD refresh rate degrades 20% when Marvell Scalable Power Management (MSPM) is enabled compared with the refresh rate when MSPM is disabled (44 fps vs. 55 fps).
- [MMP2PV-598] Video displays abnormally on an HDMI TV when mirror mode is set and VDMA is enabled.
- [MMP2PV-599] The recorded file only plays 3 seconds when recorded for 5 seconds.
- [MMP2PV-611] Video playback stops after resuming back from suspend mode.
- [MMP2PV-612] Setting the audio dual-tone multi-frequency (DTMF) fails.
- [MMP2PV-621] The displayed duration is not correct for some MP3 variable bitrate (VBR) files.
- [MMP2PV-627] The scanned application's signal strength shows incorrectly in the Wi-Fi module.
- [MMP2PV-630] In the surf browser+suspend/resume test, the suspend operation repeats several times.
- [MMP2PV-633] The CIF (352x288) size is not supported on the camcorder.
- [MMP2PV-634] The audio source microphone failed to initialize in the MediaApiTest:testSoundRecord.
- [MMP2PV-638] 720p screen size is not in SUPPORTED_SCREEN_CONFIGS in the Android Compatibility Test Suite (CTS).
- [MMP2PV-640] GB18030 encoding is UnsupportedEncoding in the CTS.
- [MMP2PV-642] The system cannot enter suspend mode while playing Flash streaming on the web.

- [MMP2PV-643] The digital rights management (DRM) player cannot display video content while audio playback works fine.
- MMP2PV-644] There is no video output on the TV path in mode 2 while the physical link with the TV is OK.
- [MMP2PV-645] The media server can crash if the system captures a still image many times in the camera.
- [MMP2PV-648] The rgb888pack/rgba888 video format flickers on the panel path graphic layer.
- MMP2PV-649] The rgb888pack/rgba888 color format does not display on the panel path graphic layer.
- [MMP2PV-652] The WI-FI driver enable/disable stress test results in a system hang (the Marvell Scalable Power Management (MSPM) is enabled).
- [MMP2PV-655] The HDMI TV set hot-plug stress test results in a 14% fail rate for no picture and no sound.
- [MMP2PV-668]The system hangs after playing a Flash advertisement in www.tudou.com.
- [CQ00129534] Gallery 3D loads images very slowly; if there are more than 100 images in one folder, some pictures require about 1 minute to load.
- [CQ00138514] Tank Recon 3D.apk does not run on the ARMADA 610 Tablet Reference Platform.
- [CQ00140959] While running Talking Tom Cat and launching Android Market, pressing the button to "add 4 extra animations" forces Talking Tom Cat to close.
- [CQ00146442] Dungeon Defenders cannot run on the ARMADA 610 Tablet Reference Platform.
- [CQ00148581] In Quadrant-3D, the earth color is wrong.
- [CQ00155941] The API Demos application (APiDemos) is not integrated on the ARMADA 610 Tablet Reference Platform, and if installed offline, OpenGL ES graphics cause the UI to hang.
- [CQ00155942] AngryBirdsSeason1.1.1.apk has small visible defects when rendering the red birds on the ARMADA 610 Tablet Reference Platform.
- [WTPSP90000011] The number of FIPS sessions cannot be more than 18 simultaneously.
- [WTPSP90000034] An Advanced Encryption Standard (AES) operation occasionally (one of tens of thousands occurrences) results in a timeout error.

5.2 Not Fixed Issues

The following issues have not been fixed in this release.

- [MMP2PV-416] A kernel error occurs when calling the ioctrl of streaming_ off after a polling call timeout occurs 3 times.
- [MMP2PV-419] The system cannot resume from suspend mode during a 2D graphics controller (GC2D) multi-thread operation.
- [MMP2PV-455] The system cannot enter a suspend mode after a resume from suspend mode stress test via a Real Time Clock (RTC) wakeup source after 2 hours.
- [MMP2PV-478] The TouchPerf test can produce about 58 events per second on the Marvell PXA920 platform, while only producing about 22 events per second on the ARMADA 610 Tablet Reference Platform.
- [MMP2PV-589] The HDMI supported resolution list read via the extended display identification data (EDID) interface is more than the actual HDMI software stack can support.
- [MMP2PV-608] Camera stress from a pre-captured still image and dumping the image to the NFS folder via USB Ethernet causes the system hang.

- [MMP2PV-610] A contact image cannot be updated by Gallery's "set as contact icon".
- [MMP2PV-617] Stress from a Read/Write SD card with 10 threads when the board repeatedly enters suspend mode and resumes causes kernel panic.
- [MMP2PV-624] The USB Command Verifier Mass Storage Class (MSC) test failed.
- [MMP2PV-629] Pushing a large file via Bluetooth fails.
- [MMP2PV-647] The libGAL.so can crash if the system captures a video clip many times in the camcorder.
- [MMP2PV-737] The kernel boot fails after accessing the SD card.
- [MMP2PV-799] The Monkey program aborts after 40 minutes of running Monkey and the stress test.
- [CQ00138997] The GPUBench-1.0.0 tool does not run on the ARMADA 610 Tablet Reference Platform.
- [CQ00147434] Using the BenchmarkSuite_v0.3_20100514.apk (from SamSung), the application auto-exits after running a while.

5.3 New Issues

The following issues are new in this release.

- [MMP2PV-248] In Mapping Buffer (mmap) mode, if the request buffer count is 3, the I/O control VIDIOC_REQBUFS returns an error.

  Workaround: Do not use mmap mode. The current camera does not support mmap mode.
- [MMP2PV-319] Video display on video layer (VL) with the GC800 performing a resize/rotation function causes an abnormal video display.
- [MMP2PV-320] The 2D color conversion displays abnormally from yuv420planar to rgb565 format.
- [MMP2PV-418] The mmap buffer mode causes a kernel problem (the board can work only after a reboot).

  Workaround: Do not use the mmap mode. The current camera does not support mmap mode.
- [MMP2PV-426] ASC II Compatible Encoding (ACE) does not work; there is no effect on the display when the ACE value is changed.
- [MMP2PV-429] Allocating more than 2 endpoints causes endpoint address conflicts.
- [MMP2PV-454] The console always outputs an I2C error after a board boot up (happened on one board).
- [MMP2PV-460] Japan FM band (76 MHz to 96 MHz) cannot be set successfully.
- [MMP2PV-512] The video display on the graphics layer/video layer (GL/VL) with the GC800 performing a rotation causes the 270 degree rotation to fail.
- [MMP2PV-548] The 2D benchmark test and suspend/resume cause the 2D benchmark test tool to exit.
- [MMP2PV-571] TCP throughput is unstable during the 200s Iperf test.
- [MMP2PV-626] Video recording and suspend/resume cause a green screen.
- [MMP2PV-662] When the android.hardware.cts.Camera_SizeTest# testConstructor calls Camera.getParameters, the test returns a NULL setting.
- [MMP2PV-672] Unsupported features, including camera, location, sensor (accelerometer), and telephony, need implementation with a reasonable no operation (No-op).
- [MMP2PV-692] Switching from the camera to the camcorder sometimes (50%) causes a green screen.

- [MMP2PV-699] The system hangs after a camera/camcorder preview of about an hour.
- MMP2PV-717] The system hangs and the graphics controller runs out of video memory after a 12-hour DVFM concurrency test.
- [MMP2PV-725] SetDisplayResolution fails with a value of GetPreferedDisplayResolution.
- [MMP2PV-744] Pushing a file to a device in the Gallery sometimes fails.
- [MMP2PV-749] The ARMADA 610 Tablet Reference Platform cannot turn on the mass storage mode on Microsoft Windows 7.
- [MMP2PV-750] The HDCP Load key always fails to load with the correct plain key and configuration file (this was not an issue in a previous release).
- [MMP2PV-767] Audio/video playback and Dynamic Voltage and Frequency Management (DVFM) cause the system to hang in less than 20 minutes.

  Workaround: Disable DDR SDRAM Auto Power Savings (APS).
- [MMP2PV-782] Some of the soft keyboard keys cannot be correctly entered.
- [MMP2PV-783] Authentication fails while sending a picture with Gallery via Bluetooth.
- [MMP2PV-784] When the "Barcode Scanner" application is opened, the screen is green.
- [MMP2PV-786] The 1080p over UYVY/YUV420/YUV422p format camera preview on the LCD is abnormal while other resolutions are okay.
- [MMP2PV-791] In the CTS, NullPointer is returned in android.media.cts.CamcorderProfileTest# testGet.
- [MMP2PV-792] A forced close exception occurred in the CTS verifier accelerometer test
- [MMP2PV-795] The camera JPEG image capture causes a JPEG header error.
- [MMP2PV-796] The HDCP Plain and Ciper Text Key load fails.
- [MMP2PV-798]The thumbnail does not match the video stream in the SD card.
- [MMP2PV-800] ADB does not work after the device reboots 310 times.
- [CQ00168056] The Asphalt_6 game sometimes auto-exits when it is running.

THIS PAGE INTENTIONALLY LEFT BLANK

A Revision History

Date	Revision	Description
July 19, 2011	C	Changes in this document: Section 1, Overview • Section 1, Overview, on page 5, removed the “Beta 1” release description in the first paragraph. • Section 1.2, Differences from the Previous Release (Beta 1), on page 6, added a description of the current release changes. • Section 1.3, Differences for Previous Releases, on page 7, added a description of the changes for other previous releases. • Table 1, ECO Information for ARMADA 610 Tablet Reference Platform, on page 5, updated Revision 4 ECO level information. Section 5, Known Issues • Section 5, Known Issues, on page 39, updated.
April 7, 2011	B	Section 1.2, Differences from the Previous Release (Beta 1), on page 6, updated.
March 28, 2011	A	Changes in this document: Section 1, Overview • Section 1, Overview, on page 5, updated the “Alpha 1” release description to “Beta 1.” • Section 1.2, Differences from the Previous Release (Beta 1), on page 6, added a description of the differences between the Alpha 1 and Beta 1 releases. • Section 1.4, Platform Features, on page 8, updated Table 2. • Section 1.5, Multimedia Features, on page 10, updated Table 3. • Section 1.6, Board Support Package Features, on page 11, added HDCP and ext4 support to the list of supported features. • Section 1.7, Release Package Contents, on page 13, updated the release package contents.

Date	Revision	Description
March 28, 2011 (cont'd)	A	Section 2, Installation • Section 2.1.1, Identifying an ARMv6 or ARMv7 Mode Boot, on page 15, added TIM update to descriptions and updated the eXtreme Debugger version required to version 5.2 Beta 1. • Section 2.1.2, Burning the Trusted WTM Firmware, OEM Boot Module, and U-Boot Binaries to eMMC Using JTAG, on page 18, revised the section with the removal of NAND support and the ddition of procedures for trusted WTM support. Updated WTM firmware version to 2.1.6. • Section 2.1.3, Burning the Non-trusted WTM Firmware, OBM, and U-Boot Binaries to eMMC Using JTAG, on page 21, updated for non-trusted WTM support. • Section 2.1.4, Burning Android on eMMC Using U-Boot, on page 24, updated section. • Section 2.2.2, Building the Source Code, on page 26, updated section. • Section 2.3, Protect the HDCP Key, on page 28, added as a new section. Section 3, Recovery and Updates This is a new section describing support for recovery and updates. Section 5, Known Issues • Section 5, Known Issues, on page 39, updated
February 24, 2011	-	Initial release.

THIS PAGE INTENTIONALLY LEFT BLANK

Marvell Semiconductor, Inc.
5488 Marvell Lane
Santa Clara, CA 95054, USA

Tel: 1.408.222.2500
Fax: 1.408.988.8279

www.marvell.com

Marvell. Moving Forward Faster