Software Test Plan

DM3730/AM3703 SOM-LV,
DM3730/AM3703 Torpedo SOM,
DM3730/AM3703 Torpedo + Wireless SOM

Android Base BSP

Revision M

History

Document Changes

The modifications that have been made to this document are listed in Table 1: History.

Date	Revision	Description	Author
1/16/2012	A	Taken from Rev A of DM3730 SOM-LV and DM3730 Torpedo SOM STP. Updates from DM37SQA-41	LN
1/31/2012	B	Updates from DM37SQA-52	LN
2/27/2012	C	Updates from DM37SQA-53	LN
3/7/2012	D	Updates from DM37SQA-55	LN
3/7/2012	E	Updates from DM37SQA-58	LN
3/19/2012	F	Updates from DM37SQA-59	LN
5/31/2012	G	Updates from DM37SQA-63	LN
6/11/2012	H	Updates from DM37SQA-65	LN
7/31/2012	I	Updates from DM37SQA-66	LN
10/03//2012	J	Updates from DM37SQA-67, DM37SQA-68, DM37SQA-69, DM3730ANDROID-347, DM37SQA-72	LN
01/04/2013	K	Updates from DM37SQA-75, DM37SQA-76, DM37SQA-77	LN
04/25/2013		Updates from DM37SQA-83 (1-5), DM37SQA-84 (1-9)	DH, LN
05/24/2013	L	Updates from DM37SQA-87 (1 - 14)	DH
06/03/2013	M	Updates from DM37SQA-87 (15-41)	DH

Table 1: History

Table of Contents

Subjects and Organization

Introduction

Purpose, Revision Control, Scope, Terminology and Acronyms, and References

Purpose

This document describes the software tests and software verification procedures for the DM37x Android Base BSP.

Regression testing of this type is statistically significant in a single-unit sample size. These tests are intended to discover repeatable flaws in the software and validate the operation of every available software feature, every communication interface, and the ability of the software to correctly interpret external stimuli.

Besides this introduction, this document is divided into two main sections: functional tests and test support appendices. The appendices are merely present for the reference of the test operator.

Revision Control

This document was prepared by and is under revision control at Logic PD. Changes to this document are to be approved by the software team leader. Should an error be discovered within this document, this document may be updated while test execution is in progress. Should this document change while test execution is in progress, some test cases may be invalidated and repeated at the discretion of the project manager.

Scope

Testing efforts described here are limited to product tests of software only. It is understood that regression testing does not exhaustively test control algorithms and the durability of hardware components. This document is intended to augment a host of other tests that validate the items outside the scope of this document.

This document is meant to be a guideline for conducting a comprehensive functional test. While it is not practical to test every software scenario, the software test outlined here will yield an adequate investigation into the integrity of software that is to be released.

Software under test (SUT) includes binary images to be used throughout testing. Consult the *Release Notes* for further information regarding specific binary images.

Terminology and Acronyms

Key terms and acronyms used in this document are listed in Table 2: Terminology.

Term	Definition
/RESET	An input pin to the microcontroller that may be used to demand a software and hardware reset
ACK/NACK	Command response: Acknowledge (ACK) / Negative Acknowledge (NACK)
DUT	Device Under Test
Errata Log	A detailed description of all “non-pass” test items, as recorded by the test operator.
JIRA	An issue tracking system used by Logic PD. (http://jira.logicpd.com/secure/BrowseProject.jspa?id=10120)
N/A	Not Applicable
Packet	A group of data sent in sequence from one node to another in a relatively short time
PC	Personal Computer
POR	Power On Reset
SUT	System Under Test
Test Progress Log	A detailed description of test progress and results as recorded by the test operator; also “Test Log"

Table 2: Terminology

Conventions

This document follows certain typographic conventions:

Convention	Description
Bold	Used in test procedures for commands, programs, and options. All terms shown in bold are typed literally.
Italic	Used in test procedures to show arguments and variables that should be replaced with user-supplied values. Italic is also used to introduce new terms, indicate filenames and directories, and to highlight comments in examples.
Bold Italic	Used in test procedures to indicate a particular key on the keyboard to be pressed.
`Constant Width`	Used in test procedures to show the contents of files or the output from commands.
`Constant Width Bold`	Used in test procedures to show commands or other text that should be typed literally by the user.
`Constant Width Italic`	Used in test procedures to show text that should be replaced with user-supplied values
`losh>`	Used in test procedures to show as the **LogicLoader** shell prompt (>).
[ ]	Square brackets orange in color with Rx-xxx enclosed are used in test procedures to indicate applicable SRS requirement tested in that particular test procedure.
//	When used following a command, text following // are comments.
< >	Used to indicate information to be replaced with user-supplied values. The <> should never be entered.
{ }	When used with text light gray in color, indicates Clear Text. Clear Text is used in test procedures to provide addition information and/or clarification.

Table 3: Conventions

References

- DM37X_Android_Torpedo_Wireless_Base_BSP_SRS.pdf

Required PC Software

- Android Debug Bridge (ADB) and supporting software (See Appendix E)

- minicom (Linux), TeraTerm (Windows) or some other serial terminal emulator program
- b/g/n wireless Ethernet card
- HP USB Formatting Tool, v2.0.6 (\\lpdsrv22\software\Tools\HP Formatter Tool 2.0.6\SP27213.exe)

Test Setup

Environment and Test Setup Diagram

Environment
All software tests should be conducted at room temperature and ambient pressure.

Test Setup
For all tests, the Debug Serial UART should be connected to a PC where you can interact with the DUT. Various tests will have you connecting other peripherals as required.

Required Equipment

General Supplies and Measurement Instruments

General Supplies

- DM3730 Development Kit, including:
  - Baseboard
  - DM3730/AM3703 Torpedo + Wireless SOM, DM3730/AM3703 Torpedo SOM, or DM3730/AM3703 SOM-LV
  - Logic PD LCD-4.3-WQVGA-10R Display
  - Logic PD LCD-4.3-WQVGA-20R Display
  - Null-modem serial cable
  - Crossover Ethernet cable
  - USB Mini-B to Standard-A cable
- Headphones or speakers with 1/8th inch stereo jack
- USB Devices
  - USB keyboard
  - USB mouse
  - USB mass storage devices (flash/thumb drives):
    - SanDisk 16 GB Cruzer USB 2.0 Flash Drive SDCZ36-016G-A11
    - SanDisk 256 MB Cruzer Mini USB 2.0 Flash Drive SDCZ2-256-A10
    - Seagate Expansion 500 GB USB 2.0 Portable External Hard Drive ST905004EXA101-RK
    - Western Digital WD Elements 1.5 TB USB 2.0 Desktop External Hard Drive
  - USB audio device (MP3 player, iPod, etc.)
  - USB webcam
  - 1.1 and 2.0 devices
  - Low, full, and high speed devices
  - Powered and unpowered hubs
  - USB Mini-B to Standard-A (male) cable (supplied in development kit)
  - USB OTG Mini-A to Mini-B *Mini-A ID Pin is grounded*
  - USB OTG Mini-A to Standard-A (female) adapter *Mini-A ID Pin is grounded*
- Two SD storage cards: one 4 GB class 6 and one 4 GB class 10 SDHC cards
- Two MMC/SD storage cards: one <=2 GB and one >=4 GB
- SD card reader/writer
- 1 GB switch (CISCO SLM2008)
- Wireless router with a/b/g/n channels
  - D-Link DGL-4500
- Aardvark I2C/SPI activity board (with connectors)
- Expansion board

Measurement Equipment

- Stopwatch or timer

Testing Procedure Requirements and Guidelines

Test Progress Reporting, Test Results, Failure Reporting

Test Progress Reporting
All test results should be recorded in written and/or electronic form within the Test Progress Log.

In some cases, undesired or unexplained operation may be detected while no specific testing is in progress. It is expected that the test engineer will record any undesired or unexplained operation in the Test Progress Log as if it were a failure that was detected during a specific test.

Tests may be repeated at any time. Test repetitions and results should be recorded in the Test Progress Log as if they had not been conducted previously.

Test Results
Each test case has its own Pass/Fail conditions. It is possible for the DUT to produce the desired outcome but fail to meet the required response time. In this event, the condition tested should be listed as a failure. Distinctions made between failures due to timing and failures due to unexpected operation should be documented in the Test Progress Log. For each test case, indicate the test result with exactly one of the following quantifiers:

Pass - Test passes without any indication of failure.

Pass R/L - Test required redlines to the test procedure to maintain a Pass result.

Concern - Test may or may not have failed due to unexpected or undesired behavior, but is not serious enough to merit a Fail rating.

Fail – All Devices Under Test (DUT) fail and may represent a significant issue.

Fail Partial – At least one DUT fails and may represent a significant issue.

Blocked – Unable to execute test, as it was prevented by some other error.

N/A – Test is not applicable. See comments for reason the test is N/A—a common reason is that the model of hardware under test does not support a feature (e.g., NOR flash).

Please note that the “Concern” quantifiers need not necessarily be linked to a specific test. If unexpected or concerning operation is witnessed between test cases, an entry in the Test Progress Log should be made. For these concerns, the test case number should be recorded as N/A.

Failure Reporting
If a test receives any result besides Pass, an issue should be logged in the Issue Tracking System (JIRA.) The new issue should include:

1. The location of the Software Test Plan in SVN.
2. The test number being executed.

3. Steps required to reproduce the test.
4. Expected behavior of the hardware/software under test.
5. Behavior of the hardware/software under test.

A reference to the new issue should be entered in the Test Progress Log.

Functional Tests

Specific Test Cases, Organized by Feature

Unless specified otherwise, no test case is dependent on another test case. The tests need not be conducted in any particular order.

Each test case must be recorded in the Test Progress Log as it is conducted.

Test 01 - Display Driver Functional Tests

Purpose

This test verifies Android display functional requirements.

Setup

Test Steps

Perform the following subtests and record results in the action log for each display as follows:

NOTE: When using LCD screens or DVI monitors, jumper JP2 on the DM3730/AM3703 Torpedo + Wireless SOM and the DM3730/AM3703 Torpedo SOM, and jumper JP5 on the DM3730/AM3703 SOM-LV must be in the ***LCD 1-2 pin position***.

NOTE: When using HDMI monitors, jumper JP2 on the DM3730/AM3703 Torpedo + Wireless SOM and the DM3730/AM3703 Torpedo SOM, and jumper JP5 on the DM3730/AM3703 SOM-LV must be in the ***HDMI 2-3 position***.

01.01 Android Display Support

01.01.01 LCD-4.3-WQVGA-20R (display 28)

[DISP-01-004] [DISP-01-005] [DISP-01-008] [SW-01-001] [SW-01-002] [UI-01-001]

Connect the display.

POR

Break into U-Boot

`DM3730Logic# setenv display 28`

List the active display: `DM3730Logic# printenv display`

`DM3730Logic# saveenv`

POR

Verify the Logic PD logo appears on an initial splash screen.

Verify the Android logo appears on a second splash screen.

Verify the Android main screen is visible without misalignment or other artifacts.

Verify the screen navigation can be done via the touch screen, without using a mouse.

01.01.02 LCD

[DISP-01-004] [DISP-01-005] [DISP-01-008] [SW-01-001] [SW-01-002] [UI-01-001]

Perform the following steps for displays 15 *LCD-4.3-WQVGA-10R display (display 15)* .

Connect the display.

POR

Break into U-Boot

`DM3730Logic# setenv display 15`

List the active display: `DM3730Logic# **printenv display**`
`DM3730Logic# **saveenv**`
POR

Verify the Logic PD logo appears on an initial splash.
Verify the Android logo appears on a second splash screen.
Verify the Android main screen is visible without misalignment or other artifacts.
Verify the screen navigation can be done via the touch screen, without using a mouse.

01.01.03 HDMI

[DISP-01-006] [DISP-01-007] [DISP-01-008]

Connect an external monitor using the HDMI cable.

On the Torpedo Launcher 2 or Torpedo Launcher 3 Baseboard, set a jumper on pin 2 and 3 of JP2. On the SOM-LV SDK2 Baseboard, set a jumper on pin 2 and 3 of JP5.

Run this test on all board configurations in vga-hdmi mode (640x480). Test the remaining resolutions [svga (800x600), xga (1024x768), 720p (1280x720)] on at least one DM3730/AM3703 Torpedo + Wireless SOM, one DM3730/AM3703 Torpedo SOM and one DM3730/AM3703 SOM-LV.

Connect the display.
POR
Break into U-Boot
`DM3730Logic# setenv display <resolution>-hdmi`
List the active display: `DM3730Logic# printenv display`
`DM3730Logic# saveenv`
`DM3730Logic# boot`

Verify the Logic PD logo appears on an initial splash.
Verify the Android logo appears on a second splash screen.
Verify the Android main screen is visible without misalignment or other artifacts.
In monitor settings, verify display resolution is correct.
NOTE: Make sure to put the jumper back in the original position (pin 1 and pin2).

01.01.04 Default Display

[DISP-01-001]

Run this test immediately after test 01.01.03 where display is set to an external monitor. If not, make sure the display is not set to display 28.

Connect the LCD-4.3-WQVGA-20R display (display 28).

POR

Break into U-Boot

```
# env default –f  (if nand booting, reset nand boot with SD card)
# saveenv
# printenv display
```

Verify the display is set to 28. (You should see something similar to 'display=28')

POR

During boot up sequence, verify display is still set to 28 and the home screen is displayed.

01.01.05 Verify OpenGL ES is running

[DISP-01-002]

POR

Open the API Demos Application

Select Graphics > OpenGL ES

Select OpenGL ES 2.0

Verify that the spinning triangle with Android logo is seen without distortion.

Push the back button

Select Touch Rotate

Rotate the cube by touching and moving your finger across the screen.

Verify the rotating cube is seen without distortion

Push the back button

Select Kube

Verify the spinning Rubik Cube is seen without distortion.

01.01.06 Verify the BSP supports hardware accelerated graphics

[DISP-01-003]

```
Android# lsmod
```

Search the output for a line similar to the following:

pvrsrvkm 158749 43 omaplfb, Live 0xbf017000

Verify that on DM3730's this line exists in the output and that on AM3703's this line does not exist in the output.

01.01.07 Verify display and graphics after suspend / resume

Open the API Demos Application

Select Graphics > OpenGL ES

Select OpenGL ES 2.0

Verify that the spinning triangle with Android logo is seen without distortion.

Turn off Stay Awake so that the DUT can suspend: **Settings > Applications > Development > Stay Awake**

Suspend the board (hit S2)

Resume the board (hit S2 again)

Open the API Demos Application

Select Graphics > OpenGL ES

Select OpenGL ES 2.0

Verify that the spinning triangle with Android logo is seen without distortion. (If the Demo application is not running do the following steps first: Select Graphics > OpenGL ES, Select OpenGL ES 2.0)

01.01.08 Portrait and Landscape modes

Go to the Display screen in Settings:

Applications > Settings > Display

Make sure that auto-rotate screen is checked. Check it if it is not.

Rotate the screen to 90 degrees:

Android# echo 90 > /cache/rotation

Verify that the screen rotates 90 degrees left of normal (portrait mode)

Rotate the screen to 270 degrees:

Android# echo 270 > /cache/rotation

Verify that the screen rotates 270 degrees left of normal (portrait mode)

Rotate the screen back to normal

Android# echo 0 > /cache/rotation

Verify that the screen is back to normal (landscape mode)

Go to the Display screen in Settings again and uncheck auto-rotate screen.

Set rotation to 90 and 270 again.

Verify that the screen did not rotate.

Suspend the SOM using the S2 button.

Resume the SOM with S2 after about 5 seconds.

Repeat the above steps (except the suspend / resume).

Verify that all portions of the test passed both before and after suspend / resume

01.02 Backlight Tests

01.02.01 Adjust backlight brightness

[DISP-01-017]

Connect the LCD-4.3-WQVGA-10R display.

POR

Press “Menu” dev kit button.

Press “Settings”

Select “Display Settings”

Select “Brightness”

Adjust dimmer to minimum level.

Verify that backlight brightness decreases.

Adjust dimmer to maximum level.

Verify that backlight brightness increases.

01.02.02 Confirm display turns on/off and dims

[DISP-01-011] [DISP-01-012] [DISP-01-013] [DISP-01-014] [DISP-01-015] [DISP-01-016] [TOUCH-01-009]

Check Stay Awake: **Settings > Applications > Development > Stay Awake**

Press “Menu” dev kit button

Select “Settings”

Select “Display”

Select “Screen timeout”

Set the screen timeout to 1 minute

Wait about 1 minute.

Verify that the display is dimmed but not turned off.

Touch the screen to wake it up.

Verify that the display is back to normal brightness.

Turn off Stay Awake so that the DUT can suspend: **Settings > Applications > Development > Stay Awake**

Suspend the board (hit S2)

Verify that the display is black

Touch the screen.

Verify that the display remains black.

Hold down the S2 button to resume.

Verify that the display is back to normal brightness.

01.02.03 Confirm display comes back when the S2 button is pushed

[DISP-01-011] [DISP-01-013] [DISP-01-014] [DISP-01-016]

Combined with test 01.02.02.

Test 02 - Audio Output Tests

Purpose

This test verifies that the functional requirements of audio output.

Setup

As preparation for this test, execute the following in sequence:

- Load any standard image
- Copy the following audio files to a desktop PC and an SD card.
  - LPCM-test-8bps.wav
  - LPCM-test-16bps.wav
  - LPCM-test-mono.wav
  - LPCM-test-stereo.wav
  - LPCM-test-{8,11,16,22,44,48}.KHz.wav
  - Any MP3 file

Test Steps

Perform the following subtests and record results in the action log:

02.01 Sound

02.01.01 Playback [AUDIO-01-001] [AUDIO-01-002]

Copy the audio files to the boot partition of the Android boot SD card. Ensure that there is at least one (1) MP3 file.
Insert the Android boot SD card.
Connect headphones or speakers to the device.
POR
Press “Apps” button
Select “Music”
Select “Unknown artist”
Select “SD card”
For each sound file:
Click sound file title
Verify that sound file sounds the same as when played back on the PC.
End For each sound file

02.01.02 Volume [AUDIO-01-003]

Press “Home” hardware button
Press “Menu”
Press “Settings”
Press “Sound”
Press “Volume”
Adjust the media slider to minimum position
Verify sound becomes softer.
Adjust the media slider to maximum position

Verify sound becomes louder.

02.01.03 Playback after suspend / resume [AUDIO-01-005]

Copy the audio files to the boot partition of the Android boot SD card.

Insert the Android boot SD card.

Connect headphones or speakers to the device.
POR

Press **Apps > Music > Grateful Dead > American Beauty > Box of Rain**

Verify that sound file sounds the same as when played back on the PC.

Turn off music.

Turn off Stay Awake so that the DUT can suspend: **Settings > Applications > Development > Stay Awake**

Suspend the board (hit S2)

Resume the board (hit S2 again)

Turn on music and play Box of Rain again.

Verify that sound file sounds the same as when played back on the PC.

Test 03 - Wired Ethernet Tests

Purpose

This test verifies that the functional requirements for wired Ethernet are met.

Setup

As preparation for this test, execute the following in sequence:

- Connect the board to an Ethernet network that has a DHCP server running.
- Determine a static IP address that can be used for these tests.
- Load any standard image

Test Steps

Perform the following subtests and record results in the action log:

03.00 Configuring Ethernet through the 'Settings' application.

03.00.01 The Ethernet device driver shall support configuration from the Android Settings application.

Test removed since requirement removed. (12/11/2012)

03.01 Configuring an IP

03.01.01 Verify boot with DHCP is successful. [ETHER-01-005] [ETHER-01-007]

POR

netcfg

Verify an IP address was obtained

Android# **ping <known host>**

Verify ping was successful

03.01.02 Verify boot with Static IP address is successful. [ETHER-01-006]

POR

In U-Boot, set kernel parameters:

For SD boot only:

env default –f (reset default environment)

For NAND boot, copy the otherbootargs from a SD boot after resetting the default environment.

End For SD boot only

setenv otherbootargs ${otherbootargs} ip=<ipaddr>:::<gatewayip>:<netmask>::

saveenv

POR

Android# **netcfg**

Verify the static IP address was obtained

Android# **ping <known host>**

Verify ping was successful.

03.01.03 Verify boot does not hang when using a Static IP address. [ETHER-01-011]

NOTE: this test is a continuation of the previous one. It assumes U-Boot environment is configured with a static IP.

POR

Record the approximate time from POR until the Android home screen appears. This is the baseline time.

Unplug the Ethernet cable.

POR

Record the approximate time from POR until the Android home screen appears. This is the second time.

Verify the second time is nearly the same as the baseline.

03.01.04 Verify boot does not hang when using a dynamic IP address. [ETHER-01-011]

POR

In U-Boot, set kernel parameters:

```
# env default -f
# saveenv
```

POR

Record the approximate time from POR until the Android home screen appears. This is the baseline time.

Unplug the Ethernet cable.

POR

Record the approximate time from POR until the Android home screen appears. This is the second time.

Verify the second time is nearly the same as the baseline.

03.01.05 Verify ifup/ifdown successfully initiates and terminates the network connection. [ETHER-01-001] [ETHER-01-003] [ETHER-01-004]

POR

```
Android# netcfg
```

Verify an IP address was obtained

```
Android# netcfg eth0 down
Android# netcfg
```

Verify the eth0 interface is listed as "DOWN".

```
Android# ping <known host>
```

Verify no ping response.

```
Android# netcfg eth0 up
Android# netcfg
```

Verify eth0 is listed as "UP".

```
Android# ping <known host>
```

Verify ping response.

03.01.06 Cross-over Private network [ETHER-01-008]

Connect PC and DUT with a cross-over cable.

Set IP of workstation: `bash$ sudo ifconfig eth0 172.20.1.1 netmask 255.0.0.0 up`

POR

Set IP of device. In U-Boot:

```
For SD boot only:
   ===> env default –f      (reset default environment)
```

For NAND boot, copy the otherbootargs from a SD boot after resetting the default environment.

```
End For SD boot only
===> setenv otherbootargs ${otherbootargs} ip=172.20.1.2::::255.0.0.0::
===> saveenv
```

POR

```
Android# netcfg
```

Verify the static IP address was obtained

Ping the Workstation from the SUT

```
Android# ping 172.20.1.1
```

Verify ping response.

Ping the SUT from the Workstation

```
Android# ping 172.20.1.2
```

Verify ping response.

Reset IP of workstation: `bash$ sudo ifconfig eth0 dhcp`

03.01.07 Verify Android Debug Bridge (ADB) over Ethernet [ETHER-01-013]

POR

Prep SUT for ADB over Ethernet

```
Android# netcfg eth0 dhcp
Android# setprop service.adb.tcp.port 5555
Android# stop adbd
Android # start adbd
```

Get SUT IP Address

```
Android# netcfg
```

On the Workstation:

```
c:\ set ADBHOST=<SUT IP Address>     (for Linux:  export ADBHOST=<SUT IP Address>)
c:\ adb kill-server
c:\ adb start-server
```

Ensure your workstation is on the same network as the DUT. At a command line on the workstation, enter the following:

```
c:\ adb devices
c:\ adb connect <SUT IP Address>
c:\ adb devices
```

Verify that the list of devices contains the DUT.

`c:\ adb –s <serial number> shell`

`(serial number is listed device IP Address + port)`

Verify that an interactive shell prompt for the remote device appears.

Type an Android command on the workstation command line:

`# netcfg`

Verify that the device displays network interface information including eth0 assigned the DUT IP address.

03.01.08 Cable events [ETHER-01-009] [ETHER-01-010]

Boot the device.

`Android# netcfg`

Ensure Ethernet is running

`Android# netcfg eth0 up`

Unplug the network cable.

Verify "`eth0: link down`" message is displayed.

Plug the cable back in.

Verify "`eth0: link up, <speed>, <duplex>`" message is displayed.

POR

03.01.09 Ethernet driver after suspend / resume [ETHER-01-015]

POR

`# netcfg`

Verify an IP address was obtained

`Android# ping <known host>`

Verify ping was successful

Turn off Stay Awake so that the DUT can suspend: **Settings > Applications > Development > Stay Awake**

Suspend the board (hit S2)

Resume the board (hit S2 again)

`Android# ping <known host>`

Verify ping was successful

On workstation:

`c:\ ping <Android ip address>`

Verify ping was successful

03.01.10 Throughput (Ethernet) [ETHER-01-012]

NOTE: Perform the following test on one board of each family (one DM3730/AM3703 Torpedo + Wireless SOM, one DM3730/AM3703 Torpedo SOM, one DM3730/AM3703 SOM-LV, etc.)

Load the Android image under test

Connect the Workstation and the SUT to the router via Ethernet cables

Put the SUT into 10BASE-T mode:

DM-37x# **netcfg eth0 up**

DM-37x# **ethtool eth0**

Verify that the DUT auto negotiated to 100MB/s (Speed: 100Mb/s)

DM-37x# **ethtool -s eth0 speed 10 duplex full autoneg off**

Verify that the DUT is at: speed = 10Mb/s

DM-37x# **ethtool eth0**

Run netperf with the SUT in transmit mode

Android # **mkdir /tmp**

Android # **netserver**

Android # **netperf –H <Linux box ip address>**

Enter the performance on the performance sheet of the TPL (netperf on Android)

Run iperf with the SUT receive mode

Linux# **mkdir /tmp**

Linux# **netperf –H <Android ip address>**

Enter the performance on the performance sheet of the TPL (netperf on Linux)

Put the SUT into 100BASE-T mode:

DM-37x# **ethtool -s eth0 speed 100 duplex full autoneg on**

Make sure that the DUT is at: speed = 100Mb/s

DM-37x# **ethtool eth0**

Run netperf with the SUT in transmit mode

Android # **netperf –H <Linux box ip address>**

Enter the performance on the performance sheet of the TPL (netperf on Android)

Run iperf with the SUT receive mode

Linux# **netperf –H <Android ip address>**

Enter the performance on the performance sheet of the TPL (netperf on Linux box)

03.02. Internet connection

03.02.01 Surf the Internet

[NONE]

Load the Android image under test

Connect the Workstation and the SUT to the router via Ethernet cables

On the homescreen, select:

Apps > Browser

Verify that the browser opens.

In the URL box, enter Yahoo.com.

Verify that the browser successfully navigates to Yahoo.com.

In the URL box, enter MSN.com.

Verify that the browser successfully navigates to MSN.com.

Test 04 - USB Host

Purpose

This test verifies that the functional requirements for USB host are met.

Setup

As preparation for this test, the following information is critical:

- Load any standard image.
- On the baseboard, ensure J42 does not have a jumper.

Test Steps

Perform the following subtests and record results in the action log:

04.01 USB Sub-System

04.01.01 HID – Keyboard / Mouse [USB-01-001] [USB-01-002]

Plug a keyboard and mouse into the USB ports.

POR

Click on the small Google search icon in the upper left corner of the home screen.

Verify a cursor moves with the mouse and that the screen changes to accept text.

Type “Android!” at the keyboard.

Verify “Android!” appears in the search bar.

Unplug keyboard and mouse, then plug them back in.

Use the keyboard’s backspace key to erase the entered text.

Verify text is erased.

If there are additional USB plugs, plug the keyboard into them and then type a few characters.

Verify what is entered on the keyboard appears on the screen.

04.01.02 HID – Keyboard / Mouse after suspend / resume [USBH-01-005]

Plug a keyboard and mouse into the USB ports.

POR

Turn off Stay Awake so that the DUT can suspend: **Settings > Applications > Development > Stay Awake**

Suspend the board (hit S2)

Resume the board (hit S2 again)

Use the mouse to click on the search bar labeled “Google.”

Verify a cursor moves with the mouse and that the screen changes to accept text.

Type “Android!” at the keyboard.

Verify “Android!” appears in the search bar.

Verify what is entered on the keyboard appears on the screen.

04.02 Mass Storage Devices

04.02.01 Mass storage device through USB ports [USBH-01-003]

- POR

- **For each from { SanDisk 16 GB Cruzer USB pendrive, SanDisk 256 MB Cruzer Mini USB pendrive, Seagate 500 GB USB 2.0 External Hard Drive,WD 1.5 TB USB External Hard Drive}:**

  - Plug the USB mass storage device into the a USB port on the DUT.

  - Mount the USB mass storage device:

    ```
    Android#  mkdir /mnt/sda
    Android# mount -t vfat /dev/block/sda1 /mnt/sda  (for device in FAT format)
    Android# mount -t ext3 /dev/block/sda /mnt/sda  (for device in EXT3 format)
    ```

  - Copy the init.rc file from the DUT to the storage device.

    ```
    Android# cat /init.rc > /mnt/sda/init_copy
    ```

  - *Verify that the file copied by using the 'cmp' command.*

    ```
    Android# cmp /init.rc /mnt/sda/init_copy
    ```

  - Copy the init_copy file back to the root

    ```
    Android# cat /mnt/sda/init_copy > /init_copy
    ```

  - *Verify that the file copied by using the 'cmp' command.*

    ```
    Android# cmp /init_copy /mnt/sda/init_copy
    Android# cmp /init_copy /init.rc
    ```

  - Remove the copy of the init_copy file from the USB mass storage device.

    ```
    Android# rm /mnt/sda/init_copy
    ```

  - Remove the copy of the init_copy file from the root.

    ```
    Android# rm /init_copy
    ```

- **End For each from {{ SanDisk 16 GB Cruzer USB pendrive, SanDisk 256 MB Cruzer Mini USB pendrive, Seagate 500 GB USB 2.0 External Hard Drive,WD 1.5 TB USB External Hard Drive }**

04.02.02 DUT as Mass storage device

- POR

- Connect the DUT to a host PC via the PC USB port and the DUT USB port.

- Verify that the PC recognizes the DUT as a mass storage device (the PC lists the DUT as a drive)

Test 05 - Serial Tests

Purpose
This test verifies that the functional requirements for serial UART support.

Setup
Load any standard image.

Test Steps
Perform the following subtests and record results in the action log:

05.01 Android Serial Port

05.01.01 Android Serial Port Support [UART-01-002]

Boot the device with any standard image.

Verify that you can read the Android terminal output.

05.01.02 Changing baud at boot

Not testing requirement. Test removed. (DM37SQA-59).

05.01.03 ttyS[0-2] [UART-01-001]

`Android#` **`busybox stty –F /dev/ttyO0`**, *verify that it returns a non-error.*

For DM3730/AM703 Torpedo SOM Only:

`Android#` **`busybox stty –F /dev/ttyO1`**, *verify that it returns a non-error.*

`(since ttO1 is dedicated to Bluetooth on DM3730/AM3703 Torpedo + Wireless SOM)`

`Android#` **`busybox stty –F /dev/ttyO2`**, *verify that it returns a non-error.*

Verify that something like the following is outputted on the Teraterm screen:

```
# busybox stty -F /dev/ttyO2
speed 9600 baud;
intr = ^C; quit = ^\; erase = ^?; kill = ^U; eof = ^D; eol = <undef>;
eol2 = <undef>; start = ^Q; stop = ^S; susp = ^Z; rprnt = ^R; werase = ^W;
lnext = ^V; flush = ^O; min = 1; time = 0;
-brkint -imaxbel
```

05.02 RS-232 Support Tests

05.02.01 Baud rates [UART-01-003]

`Android#` **`busybox stty 2400`**

Change the baud rate for the workstation terminal program to 2400.

Verify terminal is working correctly. (Type some commands, output should be readable.)

`Android#` **`busybox stty 9600`**

Change the baud rate for the workstation terminal program to 9600.

Verify terminal is working correctly. (Type some commands, output should be readable.)

```
Android# busybox stty 19200
```

Change the baud rate for the workstation terminal program to 19200.

Verify terminal is working correctly. (Type some commands, output should be readable.)

```
Android# busybox stty 38400
```

Change the baud rate for the workstation terminal program to 38400.

Verify terminal is working correctly. (Type some commands, output should be readable.)

```
Android# busybox stty 57600
```

Change the baud rate for the workstation terminal program to 57600.

Verify terminal is working correctly. (Type some commands, output should be readable.)

```
Android# busybox stty 115200
```

Change the baud rate for the workstation terminal program to 115200.

Verify terminal is working correctly. (Type some commands, output should be readable.)

05.02.02 Data character size 7,8 bits [UART-01-005]

```
Android# busybox stty cs7
```

Change the data for the workstation terminal program to 7 bits.

Verify terminal is working correctly. (Type some commands, output should be readable.)

```
Android# busybox stty cs8
```

Change the data for the workstation terminal program to 8 bits.

Verify terminal is working correctly. (Type some commands, output should be readable.)

05.02.03 Parity [UART-01-006]

```
Android# busybox stty parenb    //enable parity
Android# busybox stty parodd
```

Change the data for the workstation terminal program to odd parity.

Verify terminal is working correctly. (Type some commands, output should be readable.)

```
Android# busybox stty -parodd
```

Change the data for the workstation terminal program to even parity.

Verify terminal is working correctly. (Type some commands, output should be readable.)

```
Android# busybox stty –parenb   //no parity
```

Change the data for the workstation terminal program to no parity.

Verify terminal is working correctly. (Type some commands, output should be readable.)

05.02.04 Stop bits [UART-01-007]

```
Android# busybox stty cstopb    //two stop bits
```

Change the data for the workstation terminal program to two stop bits.

Verify terminal is working correctly. (Type some commands, output should be readable.)

```
Android# busybox stty –cstopb   //one stop bit
```

Change the data for the workstation terminal program to one stop bit.

Verify terminal is working correctly. (Type some commands, output should be readable.)

05.02.05 RTS/CTS Handshaking [UART-01-008]

```
Android# busybox stty crtscts    //enable RTS/CTS handshaking
```

Change the data for the workstation terminal program to flow control.

Verify terminal is working correctly. (Type some commands, output should be readable.)

```
Android# busybox stty –crtscts     //disable RTS/CTS handshaking
```

Change the data for the workstation terminal program to no flow control.

05.02.06 Baud rates / Serial connection after suspend / resume [UART-01-011]

POR

Turn off Stay Awake so that the DUT can suspend: **Settings > Applications > Development > Stay Awake**

Suspend the board (hit S2)

Resume the board (hit S2 again)

```
Android# busybox stty 57600
```

Change the baud rate for the workstation terminal program to 57600.

Verify terminal is working correctly. (Type some commands, output should be readable.)

```
Android# busybox stty 115200
```

Change the baud rate for the workstation terminal program to 115200.

Verify terminal is working correctly. (Type some commands, output should be readable.)

05.02.07 Determine Baud rate at run time [UART-01-004]

Combined with test 11.03.01.

05.03 DM3730/AM3703 Torpedo SOM and DM3730/AM3703 Torpedo + Wireless SOM – UART Support

05.03.01 DM3730/AM3703 Torpedo SOM and DM3730/AM3703 Torpedo + Wireless SOM- UART Support [UART-01-001]

For DM3730/AM3703 Torpedo SOM and DM3730/AM3703 Torpedo + Wireless SOM Only:

Need a db9 female to UART male connector

Connect the UART male to a standard serial female to female connector.

Connect the db9 to J25==UARTB ensuring the red line of the ribbon

cable is on the Pin1/2 side of the DM3730/AM3703 Torpedo SOM and DM3730/AM3703 Torpedo + Wireless SOM pins.

Open a terminal session on the PC to the appropriate serial port using

115200, 8-bit data, no parity, 1 stop bit.

```
Android# busybox stty 115200 < /dev/ttyO2
Android# echo "Testing /dev/ttyO2" > /dev/ttyO2
```

Verify "Testing /dev/ttyO2" is output to the terminal console POR

For DM3730/AM3703 Torpedo SOM Only:

Connect the db9 to J27==UARTA ensuring the red line of the ribbon cable is on the Pin1/2 side of the DM3730/AM3703 Torpedo SOM. Open a terminal session on the PC to the appropriate serial port using 115200, 8-bit data, no parity, 1 stop bit.

Android# **busybox stty 115200 < /dev/ttyO1**
Android# **echo "Testing /dev/ttyO1" > /dev/ttyO1**
Verify "Testing /dev/ttyO1" is output to the terminal console POR

Test 06 - MMC/SD Tests

Purpose

This test verifies that the functional requirements for MMC/SD cards are met.

Setup

As preparation for this test, execute the following:

- Create a FAT file system on an MMC/SD card if there is not one already present.
- Load any standard image into NAND.
- Boot device from NAND unless noted otherwise.

Test Steps

Perform the following subtests and record results in the action log:

06.01 Android System Support

06.01.01 Detect Insert

Boot the device from NAND. [SDMMC-01-007]

```
Android# mount
```

Ensure /mnt/sdcard is not mounted (umount if necessary)

```
Android# ls /mnt/sdcard
```

Verify that device /mnt/sdcard does not exist (/mnt/sdcard is empty)

Insert a 2GB or smaller MMC/SD card.

Verify the card was detected (a message similar to below is displayed following insertion)

```
# mmc0: new SD card at address 8000
mmcblk0: mmc0:8000 SD256 241 MiB
 mmcblk0: p1 p2
```

```
Android# ls /mnt/sdcard
```

Verify that device /mnt/sdcard exists

```
Android# dmesg
```

Review the output for a displayed message similar to the detection message following insertion.

Verify the message is displayed.

POR

06.01.02 File system [SDMMC-01-001] [SDMMC-01-002] [SDMMC-01-003] [SDMMC-01-004] [SDMMC-01-008] [SDMMC-01-011]

Boot the device from NAND.

Insert a MMC/SD card. Make sure to use some 4 GB or larger SD cards

```
Android# mount
```

Verify the SD card is mounted /dev/block/vold/179:1 /mnt/sdcard vfat... is displayed)

Create some random test data on the device:

Android# **dd if=/dev/urandom of=/testdata bs=1000 count=16000**

Copy the /testdata file to the SD card:

Android# **cat /testdata > /mnt/sdcard/testdata**

Ensure the file copied to the SD card is the same that was created (no message returned following command):

Android# **ls -l /mnt/sdcard/**

Android# **cmp /testdata /mnt/sdcard/testdata**

Remove the testdata file from the SD card and the SUT:

Android# **rm /mnt/sdcard/testdata**

Android# **rm /testdata**

POR

06.01.03 At Boot [SDMMC-01-005]

Power off device.

Insert a MMC/SD card.

Boot the device.

Android# **dmesg**

Verify the SD card was detected (message is displayed)

Android# **mount**

Verify the SD card is mounted /dev/block/vold/179:1 /mnt/sdcard vfat... is displayed)

Android# **df**

Verify the SD card is mounted (/mnt/sdcard is displayed)

06.01.04 Hardware write protect [SDMMC-01-010]

Test removed because requirement is removed.

06.01.05 High capacity cards (>4GB) [SDMMC-01-011]

Test combined with 06.01.02

06.01.06 SDIO Wireless

Plug in a SDIO wireless card.

Android# **dmesg**

Verify that it was properly detected.

Plug in a SDIO Serial card.

Android# **dmesg**

Verify that it was properly detected.

POR

06.01.07 Ready times

While watching a timer or stopwatch, quickly insert a MMC/SD card and then press Enter to run the following command.

```
Android# cat /mnt/sdcard/readme.txt
```

Verify the command returns without error within 6 seconds of inserting the MMC/SD card.

POR

06.01.08 Reinsert time

Insert a MMC/SD card.
Eject the card and reinsert it immediately.
Verify the card is recognized as inserted within 6 seconds of ejecting the card. Look for a message similar to 'mmc0: new SD card at address 0001' printed on the console.

POR

06.01.09 File system after suspend / resume [SDMMC-01-013]

Insert a MMC/SD card.
Boot the device.
Turn off Stay Awake so that the DUT can suspend: **Settings > Applications > Development > Stay Awake**

Suspend the board (hit S2)
Resume the board (hit S2 again)
Create some random test data on the device:

```
Android# dd if=/dev/urandom of=/stuff bs=1000 count=16000
```

Copy the /testdata file to the SD card:

```
Android# cat /stuff > /mnt/sdcard/stuff
Android# rm /stuff
```

Unmount and remove the SD card from the DUT
Verify that the file was written to the SD card by putting it in the workstation and checking to see that it is there.
Delete the stuff file from the SD card
POR

06.02.01 SD card access through Android GUI

Copy a .wav file and a .bmp file onto a SD card
Boot the device.
Insert the SD card
On the Android GUI:
Press "Apps" button
Select "Music"
Select "Unknown artist"

Select “SD card”
Click sound file title
Verify that the music file plays (don’t care about sound, just that it runs)
Press “Home” button
Press “Apps” icon
Select “Gallery”
Select the icon of the bitmap on the SD card
Verify that the bitmap image is shown on the screen
POR

06.03.01 Booting via 4GB class 6 and class 10 boot cards (SD card as rootfs) [SDMMC-01-006] [SDMMC-01-009]

- Create two (2) SD boot cards using the instructions from the readme instructions for the release. One boot card should be a class 6 SD card >= 4GB and the other should be a class 10 SD card >= 4GB.

- Using the class 6 boot card, boot the DUT and make sure that it is functioning i.e. interact with the DUT through the touchscreen and through Teraterm.
- Remove the SD card.
- *Verify that the DUT fails when the SD boot card is removed. This shows that the SD card is the root file system.*

- Using the class 10 boot card, boot the DUT and make sure that it is functioning i.e. interact with the DUT through the touchscreen and through Teraterm.
- Remove the SD card.
- *Verify that the DUT fails when the SD boot card is removed. This shows that the SD card is the root file system.*

Test 08 - Touch Screen Tests

Purpose

This test verifies that the functional requirements for Touch Screen for supported Logic PD displays are met.

Setup

As preparation for this test, perform the following:

- Load any standard image

Test Steps

Perform the following subtests and record results in the action log.

08.01.01 Calibrate at run time [TOUCH-01-005] [TOUCH-01-006]

For each display from (LCD-4.3-WQVGA-20R (display 28), LCD-4.3-WQVGA-10R (display 15)):

POR

Break into u-boot and set the display variable to the correct value

`OMAP Logic # setenv display <display value>`

Boot the board

On the SOM:

First, we are going to set the min and max values of the x and y coordinates to incorrect numbers to mess up the calibration:

Write the 'x_min' value into the proper file (on the SOM):

`root@android:/ # echo 100 /sys/devices/platform/omap/omap_i2c.3/i2c-3/3-0048/x_min`

Write the 'x_max' value into the proper file (on the SOM):

`root@android:/ # echo 3500 > /sys/devices/platform/omap/omap_i2c.3/i2c-3/3-0048/x_max`

Write the 'y_min' value into the proper file (on the SOM):

`root@android:/ # echo 200 > /sys/devices/platform/omap/omap_i2c.3/i2c-3/3-0048/y_min`

Write the 'y_max' value into the proper file (on the SOM):

`root@android:/ # echo 3500 > /sys/devices/platform/omap/omap_i2c.3/i2c-3/3-0048/y_max`

Stop and start the zygote to restart Android without rebooting:

```
root@android:/ # stop zygote
root@android:/ # start zygote
```

This should make the calibration off by a little bit.

Open the calculator app (touches should be off by a bit, work with it):

Home Screen > Apps Button > Calculator

Tap the screen in the four corners of the 5 button.

Verify that some of the touches trigger numbers around 5, instead of 5 (calibration is off).

Begin calibration (on SOM):

Have the SOM output touch events:

```
root@android:/ # getevent -lt /dev/input/event1
```

Empty the buffer in teraterm (or whatever you are using, if possible):

Edit > clear buffer (This should clear the screen)

Swipe a stylus from just inside the edge of the screen to off screen 4 times on each side of the screen.

Select all of the output from teraterm:

Edit > Select all (This copies everything in the buffer)

Copy the information into Excel.

In Excel:

In Excel, Select column A (make sure it gets all of it, top to bottom).

Select the “data” tab at the top of the screen.

Select “Text to Columns” on the toolbar at the top of the screen and make the information into separate columns.

In the first box in column E (or whatever the first empty column is) enter the following:

=HEX2DEC(D1) (this will copy all of the hex data in column D into column E as decimal)

Select all of the spreadsheet.

Turn on Filtering (Filter button on toolbar on “data” tab).

Filter the column that has the ABS_X and ABS_Y entries for ABS_X only.

Filter the E column (the decimal entries you created) from Smallest to Largest.

Write down the smallest ABS_X value (toss out anything funny looking i.e.1.00E+01 and any zeros).

Write down the largest ABS_X value (toss out anything funny looking i.e. any number way higher than others).

Filter the column that has the ABS_X and ABS_Y entries for ABS_Y only.

Filter the E column (the decimal entries you created) from Smallest to Largest.

Write down the smallest ABS_Y value (toss out anything funny looking i.e.1.00E+01 and any zeros).

Write down the largest ABS_Y value (toss out anything funny looking i.e. any number way higher than others).

x_min = smallest ABS_X value y_min = smallest ABS_Y value

x_max = largest ABS_X value y_max = largest ABS_Y value

On the SOM:

Write the ‘x_min’ value into the proper file (on the SOM):

```
root@android:/ # echo <x_min value> > /sys/devices/platform/omap/omap_i2c.3/i2c-3/3-0048/x_min
```

Write the ‘x_max’ value into the proper file (on the SOM):

```
root@android:/ # echo <x_min value> > /sys/devices/platform/omap/omap_i2c.3/i2c-3/3-0048/x_min
```

Write the ‘y_min’ value into the proper file (on the SOM):

```
root@android:/ # echo <y_min value> > /sys/devices/platform/omap/omap_i2c.3/i2c-3/3-0048/y_min
```

Write the ‘y_max’ value into the proper file (on the SOM):

```
root@android:/ # echo <y_max value> > /sys/devices/platform/omap/omap_i2c.3/i2c-3/3-0048/y_max
```

Stop and start the zygote to restart Android without rebooting:

```
root@android:/ # stop zygote
root@android:/ # start zygote
```

Wait for the home screen to reappear and swipe the lock.

Open the calculator app:

Home Screen > Apps Button > Calculator

Tap the screen in the four corners of the 5 button.

Verify that the touches trigger number 5 and not the numbers around it (make sure it is close, it does not have to be absolutely perfect).

End for each display from (LCD-4.3-WQVGA-20R (display 28), LCD-4.3-WQVGA-10R (display 15)):

08.01.02 Touch screen functional with USB mouse or keyboard plugged in [TOUCH-01-001] [TOUCH-01-002] [TOUCH-01-003]

For each display from (LCD-4.3-WQVGA-20R (display 28), LCD-4.3-WQVGA-10R (display 15))

Plug in a keyboard and mouse.

Verify touch screen works by navigating around a bit.

Unplug mouse, but leave keyboard plugged in.

Verify touch screen works by navigating around a bit.

Unplug keyboard, plug in mouse.

Verify touch screen works by navigating around a bit.

End for each display

08.01.03 Raw data test

Dump the raw data from the touch controller:

```
Android# hd /dev/input/event1
```

Touch the screen.

Verify that data is displayed in the terminal window.

Hit Ctl-C to exit and return to prompt

08.01.04 - Display configuration data [TOUCH-01-004]

```
Android# ls /etc/pointercal*
```

Verify that pointercal-3 [3.6”], pointercal-5 [6.4”], and pointercal-15 [4.3”] are listed.

08.01.05 Touch Screen interrupt

```
Android# cat /proc/interrupts
```

Verify interrupt #313: tsc2004 is listed

08.01.06 Sample rate [TOUCH-01-007]

```
root#android:/ # getevent -rt /dev/input/event1
```

Swipe a stylus across the screen to trigger input events.

Find the event rate for at least 20 events:

Example of output and determination of samples/second:

331-735626: 0003 0018 00000063

331-743530: 0000 0000 00000000 **rate 315**

331-746368: 0003 0001 00000861

331-746582: 0003 0000 00000f50

331-747070: 0003 0018 00000044

331-747070: 0000 0000 00000000 **rate 282**

331-750305: 0003 0001 0000081e

331-750762: 0003 0000 00000f65

331-752136: 0000 0000 00000000 **rate 197**

The rate of inputs is the average of the “rate xxx” values shown. Use a minimum of 15 “rate xxx” values to calculate the average rate of event inputs.

Verify that the rate of event inputs is at least 80 samples / second.

08.01.07 Calibration and touch after suspend / resume

POR

Using the touch screen, navigate to the Settings screen

Turn off Stay Awake so that the DUT can suspend: **Settings > Applications > Development > Stay Awake**

Suspend the board (hit S2)

Resume the board (hits S2 again)

After the suspend / resume cycle, run test 08.01.01 again (display 28 only).

Verify that touch and calibration worked correctly (08.01.01 passed) after suspend / resume.

Test 09 - SPI Tests

Purpose

This test verifies that the functional requirements for the SPI interface are met.

Setup

As preparation for this test, perform the following:

- This test is applicable to the SOM-LV only.
- Aardvark I2C/SPI Activity board setup
- Expansion board setup
  - Connect the Aardvark board to the expansion board
  - Connect the Expansion board to the DUT

Test Steps

Perform the following subtests and record results in the action log:

09.01 SPI Interface

09.01.01 SPI Test

Connect the Aardvark I2C/SPI Activity board to the expansion board as follows:
Activity board <--> Expansion board
+5V (J5.) <--> VIO_1V8 (J18.68)
GND (J5.) <--> GND (J21.1-6)
SCK (J5) <--> SPI_SCLK (J4.1)
MOSI (J5) <--> SPI TX (J4.3)
MISO (J5) <--> SPI RX (J4.5)

For chip select lines, choose:
SS (J5) <-> uP_SPI_CS0 (J4.7)
or
SS (J5) <-> uP_SPI_CS1 (J18.3)

- Connect the expansion board with Aardvark to the baseboard with the SOM-LV.

POR
- For "/dev/spidev3.0", or use uP_SPI_CS0:
- Ensure the Aardvark to Expansion board is connected as follows:
  SS (J5) <-> uP_SPI_CS0 (J4.7)
- ANDROID# **`spi-test`**
- *Verify message is displayed "SPI EEPROM Read/Write test successfull!" is displayed*
- ANDROID# **`spi-test -1`**
- *Verify message is displayed "SPI EEPROM Read/Write failed." is displayed*

- For "/dev/spidev3.1", or use uP_SPI_CS1:
- Ensure the Aardvark to Expansion board is connected as follows:
  SS (J5) <-> uP_SPI_CS1 (J18.3)
- ANDROID# **spi-test -1**
- *Verify message is displayed "SPI EEPROM Read/Write test successfull!" is displayed*
- ANDROID# **spi-test**
- *Verify message is displayed "SPI EEPROM Read/Write failed." is displayed*

09.01.02 SPI Test after suspend / resume

For "/dev/spidev3.0", or use uP_SPI_CS0:
Ensure the Aardvark to Expansion board is connected as follows:
SS (J5) <-> uP_SPI_CS0 (J4.7)
ANDROID# **spi-test**
Verify message is displayed "SPI EEPROM Read/Write test successfull!" is displayed
ANDROID# **spi-test -1**
Verify message is displayed "SPI EEPROM Read/Write failed." is displayed

Turn off Stay Awake so that the DUT can suspend: **Settings > Applications > Development > Stay Awake**

Suspend the board (hit S2)

Resume the board (hits S2 again)

For "/dev/spidev3.1", or use uP_SPI_CS1:
Ensure the Aardvark to Expansion board is connected as follows:
SS (J5) <-> uP_SPI_CS1 (J18.3)
ANDROID# **spi-test -1**
Verify message is displayed "SPI EEPROM Read/Write test successfull!" is displayed
ANDROID# **spi-test**
Verify message is displayed "SPI EEPROM Read/Write failed." is displayed

Test 10 - MTD File System Tests

Purpose

This test verifies that the functional requirements for the MTD interface are met.

Setup

As preparation for this test, perform the following:

IMPORTANT NOTE:

+ *When testing of this section has been completed, the U-Boot configuration must be restored to the default state:*
  - In LogicLoader, `losh>` **`erase /dev/nand0 B18 B4078`**
  - *Cycle power (off, then on) and let the device boot*

Test Steps

Perform the following subtests and record results in the action log:

10.02 Android File System Interface

10.02.01 NAND YAFFS2 Partition . [FLASH-01-001] [FLASH-01-003] [FLASH-01-005] [FLASH-01-006] [FLASH-01-007] [FLASH-01-008]

Load the standard image

`Android#` **`mkdir -p /mnt/mtd-nand`**

`Android#` **`cat /proc/mtd`** **`//List partitions`**

`Android#` `ls /dev/block` **`//Note the mtdblock driver number`**

`Android#` **`mount -t yaffs2 /dev/block/mtdblockx /mnt/mtd-nand`** *where x is the mtdblock driver number*

{If errors occur, restart the device and, using LogicLoader, erase NAND i.e., erase /dev/nand0 BX BY where the blocks mentioned (X for start, Y for size) span the particular NAND partition.}

`Android#` **`mount`**

Verify the yaffs2 partition was mounted: '/dev/block/mtdblock0 /mnt/mtd-nand yaffs2 …' is displayed

`Android#` **`touch /mnt/mtd-nand/newfile.txt`**

`Android#` **`ls /mnt/mtd-nand/newfile.txt`**

Verify that the file exists.

`Android#` **`rm /mnt/mtd-nand/newfile.txt`**

`Android#` **`ls /mnt/mtd-nand/newfile.txt`**

Verify that the file no longer exists.

`Android#` **`umount /mnt/mtd-nand`**

`Android#` **`mount`**

Verify the partition was unmounted (/mnt/mtd-nand is not displayed)

Remove the mtd-nand partition:

`Android#` **`rmdir /mnt/mtd-nand`**

10.02.02 NAND YAFFS2 Partition after suspend / resume

Load the standard image

Turn off Stay Awake so that the DUT can suspend: **Settings > Applications > Development > Stay Awake**

Suspend the board (hit S2)

Resume the board (hit S2 again)

```
Android# mkdir -p /mnt/mtd-nand
Android# cat /proc/mtd              //List partitions
Android# mount -t yaffs2 /dev/block/mtdblock4 /mnt/mtd-nand
```

{If errors occur, restart the device and, using LogicLoader, erase NAND i.e., erase /dev/nand0 BX BY where the blocks mentioned (X for start, Y for size) span the particular NAND partition.}

```
Android# mount
```

Verify the yaffs2 partition was mounted.

```
Android# touch /mnt/mtd-nand/newfile.txt
Android# ls /mnt/mtd-nand/newfile.txt
```

Verify that the file exists.

```
Android# rm /mnt/mtd-nand/newfile.txt
Android# ls /mnt/mtd-nand/newfile.txt
```

Verify that the file no longer exists.

```
Android# umount /mnt/mtd-nand
Android# mount
```

Verify the partition was unmounted (/mnt/mtd-nand is not displayed)

Remove the mtd-nand partition:

```
Android# rmdir /mnt/mtd-nand
```

10.02.03 Booting from NAND [FLASH-01-002] [FLASH-01-004]

Follow the instructions provided with the release to install the image into NAND.

Boot up the DUT.

Verify the DUT boots up successfully to the home screen.

10.02.04 Copying file from SD to NAND to USB stick to SD

Boot up the DUT.

Insert a SD card with file README.TXT.

Insert a USB mass storage to the USB port of the baseboard.

```
Android# mkdir -p /mnt/mtd-nand
Android# cat /proc/mtd              //List partitions
Android# mount -t yaffs2 /dev/block/mtdblock5 /mnt/mtd-nand
```

{If errors occur, restart the device and, using LogicLoader, erase NAND i.e., erase /dev/nand0 BX BY where the blocks mentioned (X for start, Y for size) span the particular NAND partition.}

Copy file from SD to NAND

```
Android# cat /mnt/sdcard/README.TXT > /mnt/mtd-nand/README.TXT
```

Copy file from NAND to USB mass storage

```
Android# mkdir /mnt/sda
```

```
Android# mount -t vfat /dev/block/sda1 /mnt/sda
```
(for device in FAT format)

```
Android# cat /mnt/mtd-nand/README.TXT > /mnt/sda/README.TXT
```

Copy file from USB mass storage to SD

```
Android# cat /mnt/sda/README.TXT > /mnt/sdcard/README2.TXT
```

Compare all of the files using the 'cmp' command:

```
Android# cmp /mnt/sdcard/README.TXT /mnt/sdcard/README2.TXT
```

```
Android# cmp  /mnt/sdcard/README2.TXT /mnt/mtd-nand/README.TXT
```

```
Android# cmp /mnt/mtd-nand/README.TXT /mnt/sda/README.TXT
```

```
Android# cmp /mnt/sda/README.TXT / mnt/sdcard/README.TXT
```

Verify the 4 files have the same contents.

10.03 NAND Testing

10.03.01 NAND Stress Test

Follow the directions in Appendix E to execute the NAND stress test.

Test 11 - System-Wide Power Management

Purpose

This test verifies that the functional requirements for power management are met.

Setup

Use any standard image.

Test Steps

Perform the following subtests and record results in the action log:

11.01 Suspend Resume

11.01.01 Enter Suspend [POWER-01-004]

Turn off Stay Awake so that the DUT can suspend: **Settings > Applications > Development > Stay Awake**

Hit the S2 button on the Device to put it into suspend Mode

Verify the device is in suspend mode.

11.01.02 Resume via S2 [POWER-01-005]

Hit the S2 button on the Device to wake it from suspend Mode

Verify the device is awake.

11.01.03 Suspend/Resume Stress Test

NOTE: The following test will take 4 – 12 hours to complete.

Turn off Stay Awake so that the DUT can suspend: **Settings > Applications > Development > Stay Awake**

Android# cat /proc/sys/kernel/printk

Android# echo 7 > /proc/sys/kernel/printk

Android# cat /sys/kernel/debug/pm_debug/wakeup_timer_seconds

Android# echo 1 > /sys/kernel/debug/pm_debug/wakeup_timer_seconds

Press S2 button to enter Suspend mode.

This script will resume the DUT from Suspend mode every second.

Depending on the availability of testing time, let the DUT run for a 4-12 hours

Verify the DUT is still running the script.

11.02 Power states

11.02.01 Power states

[POWER-01-001] [POWER-01-002][POWER-01-003][POWER-01-004]

Connect the DUT to Wattson and observe power levels throughout test.

Boot (or reboot) the DUT and login.

Verify that you are able to boot (idle state). Note the Power level of the idle state.

Turn off Stay Awake so that the DUT can suspend: **Settings > Applications > Development > Stay Awake**

At the home screen, press the globe icon to connect to the Google site.

From the Google site, navigate to some other sites (example: news article)

Verify that the Power level increases when it's loading the page(s) (run state).

Wait for a few minutes after it's done loading the page for the DUT to drop back to idle state.

Verify that the board has dropped into idle state by reduced power level.

Turn off Stay Awake so that the DUT can suspend: **Settings > Applications > Development > Stay Awake**

Wait some more, eventually the DUT will drop into suspend state.

Verify that the DUT is in suspend state by further reduced power level.

11.03 CPU Frequency

11.03.01 CPU Frequency [POWER-01-011] [POWER-01-012] [UART-01-004]

NOTE: Hardware limits for DM3730/AM3703 are 300 MHz - 1000 MHz and must be entered as Hz
cpufreq governors: conservative, ondemand, userspace, powersave, performance

POR
- Android# cd /sys/devices/system/cpu/cpu0/cpufreq

- Android# cat scaling_available_frequencies

```
Available frequencies: <AFS> = the frequencies listed as available frequencies
Invalid frequencies: <IFS> = each <AFS> + 1000
Border frequencies: <BFS> = each <AFS> - 1000
Other frequencies:  <OFS> = <0, 1>
```

- Android# cat cpuinfo_cur_freq

- Note this frequency (*Current_Frequency*) to reset the frequency back to it later.

- Android# cat scaling_governor

- Note this value (*Current_Governor*) to reset the scaling governor back to it later.

- For each from Available Frequency Steps <AFS>:

- Android# echo userspace > scaling_governor
- Android# echo <AFS> > scaling_setspeed
- Android# cat cpuinfo_cur_freq
- *Verify CPU frequency is new AFS*
- *Verify output to serial terminal is readable*
- Android# cat stats/time_in_state
- *Verify transition time in state for AFS*
- Android# cat stats/time_in_state
- *Verify transition time in state has been updated*

- End For each

- Android# echo *Current_Frequency* > scaling_setspeed
- Android# cat cpuinfo_cur_freq
- *Verify CPU frequency is Current_Frequency*

- For each from Invalid Frequency Steps <IFS>:
  - Android# echo <IFS> > scaling_setspeed
  - Android# cat cpuinfo_cur_freq
  - *Verify that the current frequency is the frequency entered, rounded up to the nearest valid <AFS> frequency, unless the <IFS> was higher than the highest <AFS> frequency, in which case it rounds down.*
- End For each

- Android# echo *Current_Frequency* > scaling_setspeed
- Android# cat cpuinfo_cur_freq
- *Verify CPU frequency is Current_Frequency*

- For each from Border Frequency Steps <BFS>:
  - Android# echo <BFS> > scaling_setspeed
  - Android# cat cpuinfo_cur_freq
  - *Verify that the new frequency is the entered <BFS> frequency rounded up to the nearest <AFS> frequency.*
- End For each

- For each from Other Frequency Steps <OFS>:
  - Android# echo <OFS> > scaling_setspeed
  - Android# cat cpuinfo_cur_freq
  - *Verify that the new frequency the lowest <AFS>.*
- End For each

- Android# echo *Current_Frequency* > scaling_setspeed
- Android# cat cpuinfo_cur_freq

- *Verify CPU frequency is Current_Frequency*

Find out what governers are available:

- Android# cat scaling_available_governors

For each <governor> from available governors:

- Android# **`echo <governor> > scaling_governor`**
- Android# `cat scaling_governor`
- *Verify governor is <governor>*

End For each <governor> from available governors:

- Android# `echo Current_Governor > scaling_governor`
- Android# `cat scaling_governor`
- *Verify governor is Current_Governor*

11.03.02 SmartReflex [POWER-01-013]

- Connect the DUT to Wattson and observe power levels throughout test.
- Boot (or reboot) the DUT and login.
- Note the power level of the SOM main battery from Wattson
- Turn off SmartReflex:

  -Android# echo 0 > /sys/kernel/debug/smartreflex/sr_core/autocomp

  -Android# echo 0 > /sys/kernel/debug/smartreflex/sr_mpu_iva/autocomp

- Observe the power level of the SOM main battery
- *Verify that the power consumption of the SOM increased*
- Turn on SmartReflex:

  -Android# echo 1 > /sys/kernel/debug/smartreflex/sr_core/autocomp

  -Android# echo 1 > /sys/kernel/debug/smartreflex/sr_mpu_iva/autocomp

- Observe the power level of the SOM main battery
- *Verify that the power consumption of the SOM decreased*

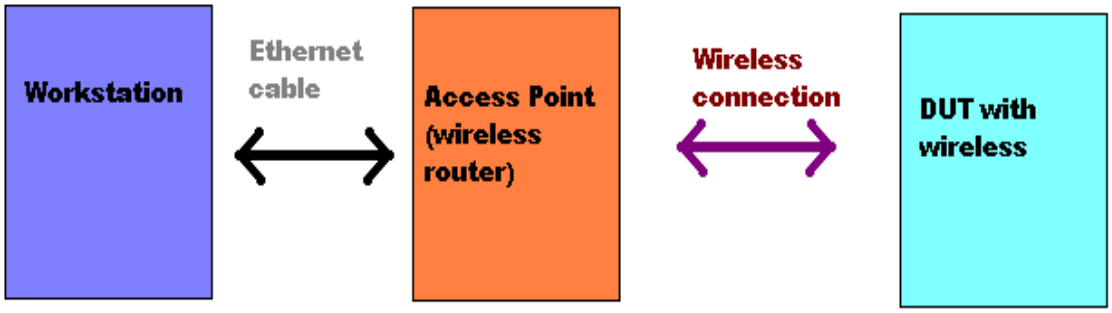
Test 12 - Wireless Ethernet Tests

Purpose

This test verifies the functionality of the wireless support.

Setup

The following figure shows the recommended test setup.



* SQA uses a wireless router that is approximately 1 – 3 feet from the DUTs being tested, and may or may not have a single cubicle wall in between the router and the DUTs.

Testing Tools

The following tools are required for testing:

- Netperf: A traffic generator. The Netperf tool generates data at a given rate. It runs as an embedded component on the host. Netperf supports transport Layer 4 TCP and UDP protocols.
- An access point (AP i.e. a router).
- One workstation (STA).
- A DUT to test wireless on

Test Steps

Perform the following subtests and record results in the action log:

IMPORTANT: **Attach wireless antenna to the right socket on the SOM.**

IMPORTANT: **Make sure that the wired Ethernet cable is disconnected during Wi-Fi testing.**

NOTE: You may need to turn off your workstation's firewall to successfully ping it from the DUT.

12.01 Wireless Connection

12.01.01 Open Wireless Connection [WIFI-01-005]

POR

For each from: {Wireless A, B, G, N mode}

Set up the wireless router with the following:

- Wireless Network Name (SSID): SQATest
- Choose appropriate band:
  - o 2.4GHz for B, G, N mode
  - o 5GHz for A or N mode
- Security Mode: None

In Android, navigate to: **Settings => Wireless & Networks => Wi-Fi Settings**

Turn on **Wi-Fi** if not already

Turn on **Network Notification** if not already

*Verify **SQATest** is listed under the **Wi-Fi networks** section*

Press **SQATest**

Press **Connect**

Verify **Settings => Wireless & Networks => Wi-Fi Settings** *=> **Wi-Fi** is connected to **SQATest***

Ping the workstation (or other known IP address) from the DUT:

```
Android# ping <workstation IP address>
```

Verify the ping replies successfully

Get the DUT IP address:

Navigate to **Settings => Wireless & Network => Wi-Fi Settings**

Press the **Menu** key

Press the **Advanced** button on the screen

Verify the MAC address is a Logic PD MAC address (it starts with 00:08:EE:…)

The IP address is listed on the screen

Ping the DUT from work station:

```
c:\ ping <DUT IP address>
```

Verify the ping replies successfully

NOTE: Execute the following steps on one SOM per family (one DM3730/AM3703 Torpedo + Wireless SOM, one DM3730/AM3703 Torpedo SOM, one DM3730/AM3703 SOM-LV, etc.)

Run netperf with the DUT in transmit mode:

```
Android # mkdir /tmp

Android # netserver

Android # netperf –H <Linux box ip address>
```

When netperf begins, hit the reset button on Wattson. Do an Alt+Print Screen before the netperf stops (about 5 seconds) to copy the results from Wattson, then paste them into Paint and make a cut/copy of just the "Monitor" results

```
Enter the power results on the performance sheet of the TPL

Enter the throughput performance on the performance sheet of the TPL (netperf on Android)
```

Run netperf with DUT in receive mode:

```
Linux# mkdir /tmp

Linux# netserver

Linux# netperf –H <DUT ip address>
```

When netperf begins, hit the reset button on Wattson. Do an Alt+Print Screen before the netperf stops (about 5 seconds) to copy the results from Wattson, then paste them into Paint and make a cut/copy of just the "Monitor" results

```
Enter the power results on the performance sheet of the TPL

Enter the throughput performance on the performance sheet of the TPL (netperf on Linux box)
```

End for each from: {Wireless A, B, G, N mode}

12.01.02 WEP Wireless Connection [WIFI-01-008] [WIFI-01-011]

NOTE: THE JUMPER JP6 ON THE SDK2 BASEBOARD *MUST* BE IN THE 2-3 DGND POSITION AND NOT THE 1-2 5V POSITION WHILE USING WATTSON OR THE READINGS WILL BE INCORRECT

For each from: {Wireless A, B, G mode}

Set up the wireless router with the following:

- Wireless Network Name (SSID): SQATest
- Choose appropriate band:
  - 2.4GHz for B, G mode
  - 5GHz for A mode
- Security Mode: WEP
- WEP Key 1: ABCD567890 (or whatever appropriate)

POR (optional)

In Android, navigate to: **Settings => Wireless & Networks => Wi-Fi Settings**

Turn on **Wi-Fi** if not already

Turn on **Network Notification** if not already

Verify ***SQATest*** *is listed under the* ***Wi-Fi networks*** *section*

Press **SQATest**

Enter password “ABCD567890” (or whatever was entered as the WEP Key) via the on-screen keyboard or via the terminal:

```
Android# input text “ABCD567890”
```

Press **Connect**

Verify **Settings => Wireless & Networks => Wi-Fi Settings** *=>* ***Wi-Fi*** *is connected to* ***SQATest***

Ping the workstation (or other known IP address) from the DUT:

```
Android# ping <workstation IP address>
```

Verify the ping replies successfully

Get the DUT IP address:

Navigate to **Settings => Wireless & Network => Wi-Fi Settings**

Press the **Menu** key

Press the **Advanced** button on the screen

The IP address is listed on the screen

Ping the DUT from work station:

```
c:\ ping <DUT IP address>
```

Verify the ping replies successfully

Remove the wireless connection:

Press **SQATest**, then press **Forget**

End for each from: {Wireless A, B, G mode}

12.01.03 WPA and WPA2 Wireless Connection [WIFI-01-011]

For each from: {Wireless A, B, G, N mode}

Set up the wireless router with the following:

- Wireless Network Name (SSID): SQATestWPA
- Choose appropriate band:
  - o 2.4GHz for B, G, N mode
  - o 5GHz for A or N mode
- Security Mode: WPA / AES
- WPA Key: ABCD567890 (or whatever appropriate)

POR (optional)

In Android, navigate to: **Settings => Wireless & Networks => Wi-Fi Settings**

Turn on **Wi-Fi** if not already

Turn on **Network Notification** if not already

Verify ***SQATestWWPA*** *is listed under the* ***Wi-Fi networks*** *section*

Press **SQATestWPA**

Enter password “ABCD567890” (or whatever was entered as the WPA Key) via the on-screen keyboard or via the terminal:

```
Android# input text “ABCD567890”
```

Press **Connect**

Verify **Settings => Wireless & Networks => Wi-Fi Settings** *=>* ***Wi-Fi*** *is connected to* ***SQATestWPA***

Ping the workstation (or other known IP address) from the DUT:

```
Android# ping <workstation IP address>
```

Verify the ping replies successfully

Ping the DUT from work station:

(Note: DUT IP address is listed under Settings > Wireless & Network > Wi-Fi Setting. Press the **Menu** key. Press the **Advanced** button on the screen.)

```
c:\ ping <DUT IP address>
```

Verify the ping replies successfully

Remove the wireless connection:

Press **SQATestWPA**, then press **Forget**

End for each from: {Wireless A, B, G, N mode}

For each from: {Wireless A, B, G, N mode}

Set up the wireless router with the following:

- Wireless Network Name (SSID): SQATestWPA2
- Choose appropriate band:
  - o 2.4GHz for B, G, N mode
  - o 5GHz for A or N mode
- Security Mode: WPA2 / AES
- WPA2 Key: ABCD567890 (or whatever appropriate)

POR (optional)

In Android, navigate to: **Settings => Wireless & Networks => Wi-Fi Settings**

Turn on **Wi-Fi** if not already

Turn on **Network Notification** if not already

Verify ***SQATestWWPA2*** *is listed under the* ***Wi-Fi networks*** *section*

Press **SQATestWPA2**

Enter password “ABCD567890” (or whatever was entered as the WPA2 Key) via the on-screen keyboard or via the terminal:

```
Android# input text “ABCD567890”
```

Press **Connect**

Verify **Settings => Wireless & Networks => Wi-Fi Settings** *=>* ***Wi-Fi*** *is connected to* ***SQATestWPA2***

Ping the workstation (or other known IP address) from the DUT:

```
Android# ping <workstation IP address>
```

Verify the ping replies successfully

Ping the DUT from work station:

(Note: DUT IP address is listed under Settings > Wireless & Network > Wi-Fi Setting. Press the **Menu** key. Press the **Advanced** button on the screen.)

```
c:\ ping <DUT IP address>
```

Verify the ping replies successfully

Remove the wireless connection:

Press **SQATestWPA2**, then press **Forget**

End for each from: {Wireless A, B, G, N mode}

12.01.04 Static IP [WIFI-01-004]

POR

Following the directions in previous tests, set up a wireless connection with **SQATest** in either open or WEP security.

Ping the DUT from the workstation:

```
c:\ping <DUT dynamic IP address>
```

Verify that the ping replies successfully.

Navigate to **Settings > Wireless & Networks > Wi-Fi Settings**

Press the Menu key

Press **Advanced**

Select **Use Static IP** and enter the network settings (IP address, gateway, netmask)

```
Ping the DUT from the workstation:
```

```
c:\ping <DUT dynamic IP address>
```

Verify that there is no reply.

```
c:\ping <DUT static IP address>
```

Verify that the ping replies successfully.

```
Ping the workstation from the DUT:
```

```
Android# ping <workstation IP address>
```

Verify **Settings => Wireless & Networks => Wi-Fi Settings** *=> **Wi-Fi** is connected to **SQATest***

12.01.05 Wireless functionality after suspend / resume. [WIFI-01-001] [WIFI-01-002] [WIFI-01-003] [WIFI-01-013]

POR

Following the directions in previous tests, setup a wireless connection with **SQATest** in either open or WEP security.

Ping the workstation from the DUT:

```
Android# ping <workstation IP address>
```

Verify that the ping replies successfully.

Bring the wireless network down:

```
Android# netcfg wlan0 down
```


Verify **Settings => Wireless & Networks => Wi-Fi Settings** *=>* ***Wi-Fi*** *is disconnected*

Bring the wireless network up:

```
Android# netcfg wlan0 up
```

Verify **Settings => Wireless & Networks => Wi-Fi Settings** *=>* ***Wi-Fi*** *is connected to* ***SQATest***

Turn off Stay Awake so that the DUT can suspend.

Settings > Applications > Development > Stay Awake

Suspend the board (hit S2)

Verify that the board has transitioned to Suspend state.

Resume the board (hit S2 again)

Verify that the ping resumes after the board wakes up.

Verify **Settings => Wireless & Networks => Wi-Fi Settings** *=>* ***Wi-Fi*** *is connected to* ***SQATest***

12.01.06 Web Browsing via Wireless Network Connection

POR

Remove the Ethernet cable to ensure wireless network is being tested.

Connect to a wireless network (Utica-Dev is recommended. For the key/password information, see “Logic Wireless” on the intranet.)

Once connected, enter the URL of a known internet site (ex: www.logicpd.com, www.yahoo.com)

Verify the DUT successfully connects to the URL entered above.

12.02 Wireless and Bluetooth

12.02.01 Wi-fi and Bluetooth On Concurrently

POR

Following the directions in previous tests, setup a wireless connection with **SQATest** in either open or WEP security.

Ping the workstation from the DUT:

```
Android# ping <workstation IP address>
```

Verify that the ping replies successfully.

Ping the DUT from work station:

```
c:\ ping <DUT IP address>
```

Verify the ping replies successfully.

In Android, navigate to: **Settings** => **Wireless & Networks**

Turn on **Bluetooth** if not already done so.

Select **Bluetooth Settings.**

Make the DUT discoverable by selecting **Discoverable**.

From the “known good” device, scan for devices.

Verify that the DUT is listed along with any nearby devices.

Select the DUT to pair it with the “known good” device.

Verify that the two devices are paired.

Send a picture from the “known good” device to the DUT via Bluetooth. If prompted, confirm the DUT is ready to receive the file.

Verify that the picture is successfully sent to the DUT.

Verify that the DUT and workstation still receive ping replies from each other.

Test 13 - Bluetooth Tests

Purpose

This test verifies that the functional requirements for Bluetooth interface are met.

Setup

As preparation for this test, execute the following in sequence:

- Load any standard image
- Attach wireless antenna to socket J3 on SOM.
- Prepare a Bluetooth-capable device (Laptop, Bluetooth adapter, etc.) with known Bluetooth address. This is called a “known good” device.

Test Steps

Perform the following subtests and record results in the action log:

13.04 Data transfer with a remote Bluetooth device

13.04.02 DUT sending file to Bluetooth device [BT-01-001] [BT-01-002] [BT-01-003] [BT-01-004]

In Android, navigate to: **Settings** => **Wireless & Networks**

Turn on **Bluetooth** if not already done so

Select **Bluetooth Settings**

Make the “known good” device discoverable, if necessary.

Select Scan.

Verify that the “known good” device is listed along with any nearby devices.

Select the “known good” device to pair it with the DUT.

Verify that the two devices are paired.

Select a picture from the Gallery and send it via Bluetooth to the “known good” device. (The picture can be on a SD card.)

Verify that the picture is successfully sent to the “known good” device.

Open the picture and verify that it is the same picture that was sent from the DUT.

13.04.03 DUT receiving file from Bluetooth device [BT-01-001] [BT-01-002] [BT-01-003] [BT-01-004]

In Android, navigate to: **Settings** => **Wireless & Networks**

Turn on **Bluetooth** if not already done so.

Select **Bluetooth Settings.**

Make the DUT discoverable by selecting **Discoverable**.

From the “known good” device, scan for devices.

Verify that the DUT is listed along with any nearby devices.

Select the DUT to pair it with the “known good” device.

Verify that the two devices are paired.

Send a picture from the “known good” device to the DUT via Bluetooth. If prompted, confirm the DUT is ready to receive the file.

Verify that the picture is successfully sent to the DUT.

Open the picture and verify that it is the same picture that was sent from the “known good” device.

13.05 Support customers to program and build Bluetooth applications.

13.05.01 Verify the Bluez library and utilities built. [BT-01-005]

Test removed.

13.05.02 Verify the Bluetooth demo application performs successfully [BT-01-001]

Test removed.

13.06 Bluetooth support after suspend / resume

13.06.01 Verify the Bluetooth interface can be successfully started after suspend / resume

In Android, navigate to: **Settings** => **Wireless & Networks**

Turn on **Bluetooth** if not already done so.

Select **Bluetooth Settings.**

Make the DUT discoverable by selecting **Discoverable**.

From a 2nd device (can be another mobile device or another DUT), scan for devices.

Verify that the DUT is listed along with any nearby devices.

Select the DUT to pair it with the 2nd device.

Verify that the two devices are paired.

Turn off Stay Awake so that the DUT can suspend: **Settings > Applications > Development > Stay Awake**

Suspend the board (hit S2)

Resume the board (hit S2 again)

Navigate to the **Bluetooth Settings** screen.

Verify the DUT is paired with the 2nd device.

Send a picture from the DUT to the 2nd device via Bluetooth. If prompted, confirm the DUT is ready to receive the file.

Verify that the picture is successfully sent to the 2nd device.

Open the picture and verify that it is the same picture that was sent from the 1st DUT.

Test 15 - USB OTG Tests

Purpose

This test verifies that the functional requirements for USB OTG host and USB OTG device are met.

Setup

As preparation for this test, the following information is critical:

- Any standard image may be loaded
- On the baseboard, ensure a jumper is not placed at J42.
- For all OTG tests, always insert and remove the USB Device AND the adapter from the USB OTG port.

For USB OTG Host tests:

- A USB OTG Mini-A to Mini-B cable plugged into the device at boot. The Mini-A end should be plugged in to the device. An “A” is embossed on the Mini-A connector end of the cable. The Mini-B connector end of the cable may or may not be connected to a device.
- A USB OTG Mini-A to Standard-A (female) adapter may be used, and must also be plugged into the device at boot. The Mini-A end should be plugged in to the device. The Standard-A connector end of the adapter may or may not be connected to a device.
- Logic PD uses a USB OTG Mini-A to Standard-A (female) adapter

- For USB OTG device tests:
  - Logic PD supplies a USB Mini-B to Standard-A (male) cable as part of the development kit. This cable should be used when testing USB OTG device mode. It may or may not be plugged in at boot.

NOTE: Any devices that use more than 100mA must be externally powered since the USB OTG port can only provide 100mA.

Perform the following subtests and record results in the action log:

15.01 USB OTG Host-Mode Tests – USB Sub-System

15.01.01 HID – Keyboard/Mouse [USBOTG-01-001] [USBOTG-01-002]

Plug the USB OTG Mini-A to Standard-A (female) adapter into the USB OTG port.

Plug a USB mouse into the USB OTG Mini-A to Standard-A (female) adapter

POR

Click on the small Google search icon in the upper left corner of the home screen.

Verify a cursor moves with the mouse and that the screen changes to accept text.

Unplug mouse and plug a USB keyboard into the USB OTG Mini-A to Standard-A (female) adapter

Type “Android!” at the keyboard.

Verify “Android!” appears in the search bar.

Unplug keyboard, and plug USB mouse into the USB OTG Mini-A to Standard-A (female) adapter

Use the mouse cursor to press Go.

Verify that interface responds.

15.01.02 HID – Keyboard/Mouse after suspend / resume [USBOTG-01-005]

Plug the USB OTG Mini-A to Standard-A (female) adapter into the USB OTG port.

Plug a USB mouse into the USB OTG Mini-A to Standard-A (female) adapter

POR

Turn off Stay Awake so that the DUT can suspend: **Settings > Applications > Development > Stay Awake**

Suspend the board (hit S2)

Resume the board (hit S2 again)

Use the mouse to click on the search bar labeled “Google.”

Verify a cursor moves with the mouse and that the screen changes to accept text.

Unplug mouse and plug a USB keyboard into the USB OTG Mini-A to Standard-A (female) adapter

Type “Android!” at the keyboard.

Verify “Android!” appears in the search bar.

Unplug keyboard, and plug USB mouse into the USB OTG Mini-A to Standard-A (female) adapter

Use the mouse cursor to press Go.

Verify that interface responds.

15.01.03 Mass storage device through USB ports [USBOTG-01-002]

- POR

- **For each from { SanDisk 16 GB Cruzer USB pendrive, SanDisk 256 MB Cruzer Mini USB pendrive, Seagate 500 GB USB 2.0 External Hard Drive,WD 1.5 TB USB External Hard Drive}:**
  - Plug the USB mass storage device into the a USB OTG port on the DUT.

    (Note: If the USB device is drawing more than 100mA, it must be externally powered.)
  - Mount the USB mass storage device:

    ```
    Android#  mkdir /mnt/sda
    Android# mount -t vfat /dev/block/sda1 /mnt/sda  (for device in FAT format)
    Android# mount -t ext3 /dev/block/sda1 /mnt/sda  (for device in EXT3 format)
    ```
  - Copy the init.rc file from the DUT to the storage device:

    ```
    Android# cat /init.rc > /mnt/sda/init_copy
    ```


- Verify that the file copied by using the ‘cmp’ command:

```
Android# cmp /init.rc /mnt/sda/init_copy
```

- Copy the init_copy file back to the root:

```
Android# cat /mnt/sda/init_copy > /init_copy
```

- Verify that the file copied by using the ‘cmp’ command:

```
Android# cmp /init_copy /mnt/sda/init_copy
```

- Verify that /init.rc is the same as /init_copy using the ‘cmp’ command:

```
Android# cmp /init_copy /init.rc
```

- Remove the copy of the init_copy file from the USB mass storage device:

```
Android# rm /mnt/sda/init_copy
```

- Remove the copy of the init_copy file from the root:

```
Android# rm /init_copy
```

- End For each from {{ SanDisk 16 GB Cruzer USB pendrive, SanDisk 256 MB Cruzer Mini USB pendrive, Seagate 500 GB USB 2.0 External Hard Drive,WD 1.5 TB USB External Hard Drive }

15.02 USB Device-mode Tests

15.02.01 Verify Android Debug Bridge (ADB) over USB [USBOTG-01-003]

Remove the USB OTG Mini-A to Standard-A (female) adapter and connect the DUT to the workstation with a USB cable via the USB OTG port.
Remove the Ethernet cable from the SUT to ensure ADB runs over USB OTG

Prep SUT for ADB over USB:

```
Android# stop adbd
Android# start adbd
```

On the Workstation:

```
c:\adb kill-server
c:\adb start-server
```

At a command line on the workstation, enter the following:

```
c:\adb devices
```

Verify that the list of devices contains the DUT.

```
c:\adb –s <device number> shell
     (device number is listed under adb devices)
```

Verify that an interactive shell prompt for the remote device appears.

Type an Android command:

```
Android# ls
```

Verify that the right contents are displayed.

15.02.02 DUT as Mass storage device

- POR
- Connect the DUT to a host PC via the PC USB port and the DUT USB OTG port.
- Verify that the PC recognizes the DUT as a mass storage device (the PC lists the DUT as a drive)

15.02.03 ADB (Ethernet over USBOTG) - USB OTG mass storage devices - ADB (Ethernet over USBOTG)

Connect the DUT to the workstation with a USB cable via the USB OTG port.
Remove the Ethernet cable from the SUT to ensure ADB runs over USB OTG.

Prep SUT for ADB over USB:

```
Android# stop adbd
Android# start adbd
```

On the Workstation:

```
c:\adb kill-server
c:\adb start-server
```

At a command line on the workstation, enter the following:

```
c:\adb devices
```

Verify that the list of devices contains the DUT.

```
c:\adb –s <device number> shell
     (device number is listed under adb devices)
```

Verify that an interactive shell prompt for the remote device appears.

Type an Android command:

```
Android# ls
```

Verify that the right contents are displayed.

Remove the USB Standard-A to Mini-A

Using one of the USB mass storage devices in the SQA set:
Plug the USB mass storage device into the USB OTG port on the DUT (using an adapter).
(Note: If the USB device is drawing more than 100mA, it must be externally powered.)
Mount the USB mass storage device:

```
Android# mkdir /mnt/sda
Android# mount -t vfat /dev/block/sda1 /mnt/sda  (for device in FAT format)
Android# mount -t ext3 /dev/block/sda1 /mnt/sda  (for device in EXT3 format)
```

Cat the README.TXT file on the mass storage deivce

```
Android# cat /mnt/sda/README.TXT
```

Verify that the output of the 'cat' command is as expected.

Remove the USB mass storage device and USB OTG adapter.

Reconnect the DUT to the workstation with a USB cable via the USB OTG port.

Prep SUT for ADB over USB:

```
Android# stop adbd
Android# start adbd
```

On the Workstation:

```
c:\adb kill-server
c:\adb start-server
```

At a command line on the workstation, enter the following:

```
c:\adb devices
```

Verify that the list of devices contains the DUT.

```
c:\adb –s <device number> shell
    (device number is listed under adb devices)
```

Verify that an interactive shell prompt for the remote device appears.

Type an Android command:

```
Android# ls
```

Verify that the right contents are displayed.

Test 16 – General Tests

Purpose

This test verifies that the functional requirements for general features such as Real Time Clock (RTC), Product ID and GPIO are met.

Setup

As preparation for this test, the following information is critical:

- Any standard image may be loaded

Perform the following subtests and record results in the action log:

16.01 Real Time Clock (RTC)

16.01.01 Time transitions from am to pm, Day, Month & Year remain the same [GEN-01-001] [GEN-01-002]

POR

In the GUI interface, select Apps > Settings > Date & Time

Touch the time in the upper right hand corner of the screen to see the date in the upper left hand corner of the screen.

Turn off (if not already) the Automatic time set.

Select > Set date

Set the date to 01/01/2011

Select > Set time

Set the time to 11:59

Wait for a minute or so and see that in the upper right corner of the screen, the time changes to 12:00

Verify that the time has transitioned from am to pm (time changed to 12:00)

16.01.02 Time transitions from pm to am, Day changes, Month & Year remain the same [GEN-01-001] [GEN-01-002]

In the GUI interface, select Apps > Settings > Date & Time

Turn off (if not already) the Automatic time set.

Select > Set time

Set the time to 23:59

Wait for a minute or so and see that in the upper right corner of the screen, the time changes to 00:00

Verify that the time has transitioned from pm to am (time changed to 00:00)

Verify that the date has changed to 01/02/2011

16.01.03 Time transitions from pm to am, Day, Month & Year change [GEN-01-001] [GEN-01-002]

In the GUI interface, select Apps > Settings > Date & Time

Turn off (if not already) the Automatic time set.

Select > Set date

Set the date to 12/31/2011

Select > Set time

Set the time to 23:59

Wait for a minute or so and see that in the upper right corner of the screen, the time changes to 12:00

Verify that the time changed to 00:00.

Verify that the date changed to 01/01/2012

16.01.04 Non-Leap year [GEN-01-001] [GEN-01-002]

In the GUI interface, select Apps > Settings > Date & Time

Turn off (if not already) the Automatic time set.

Select > Set date

Set the date to 02/28/2011

Select > Set time

Set the time to 23:59

Wait for a minute or so and see that in the upper right corner of the screen, the time changes to 00:00

Verify that the time changed to 00:00

Verify that the date changed to 03/01/2011

16.01.05 Leap year [GEN-01-001] [GEN-01-002]

In the GUI interface, select Apps > Settings > Date & Time

Turn off (if not already) the Automatic time set.

Select > Set date

Set the date to 02/28/2012

Select > Set time

Set the time to 23:59

Wait for a minute or so and see that in the upper right corner of the screen, the time changes to 00:00

Verify that the time changed to 00:00

Verify that the date changed to 02/29/2012

16.01.06 Power cycle while power connected [GEN-01-001] [GEN-01-002]

In the GUI interface, select Apps > Settings > Date & Time

Turn off (if not already) the Automatic time set.

Select > Set date

Set the date to 05/05/2005

Select > Set time

Set the time to 05:05

Hit the reset button (S1)

When the board comes back up:

Verify that the time is what it should be (a little later than 05:05)

Verify that the date did not change (05/05/05)

16.01.07 Power cycle while power + power disconnect [GEN-01-001] [GEN-01-002] [GEN-01-003]

*NOTE: This test is for the DM3730/AM3703 Torpedo SOM and DM3730/AM3703 Torpedo + Wireless SOM only. The DM3730/AM3703 SOM-LV does not have a backup battery.

In the GUI interface, select Apps > Settings > Date & Time
Turn off (if not already) the Automatic time set.
Select > Set date
Set the date to 05/05/2005
Select > Set time
Set the time to 05:05
Power down the board and unplug the power supply from the board.
Wait 2 minutes
Plug the power supply back into the board and power the board back up.
Verify that the time is what it should be (a little later than 05:05)
Verify that the date did not change (05/05/05)

16.01.08 Setting time and transition after suspend / resume

POR
Turn off Stay Awake so that the DUT can suspend: **Settings > Applications > Development > Stay Awake**

Suspend the board (hit S2)
Resume the board (hit S2 again
In the GUI interface, select Apps > Settings > Date & Time
Touch the time in the upper right hand corner of the screen to see the date in the upper left hand corner of the screen.
Turn off (if not already) the Automatic time set.
Select > Set date
Set the date to 01/01/2011
Select > Set time
Set the time to 11:59
Wait for a minute or so and see that in the upper right corner of the screen, the time changes to 12:00
Verify that the time has transitioned from am to pm (time changed to 12:00)

16.02 Product ID Information

16.02.01 Physical verification of product ID info [GEN-01-004]

Break into U-Boot and examine the product ID information output in the first few lines.
Verify DRAM matches the hardware configuration.
Verify NAND matches the hardware configuration.
Verify Board matches the hardware configuration.

Verify Part Number matches the hardware configuration.
Verify Model Name matches the hardware configuration.
Verify Serial Number matches the hardware configuration.

16.02.02 Verification of software access to product ID info [GEN-01-004] [GEN-01-006]

POR

```
Android# cd /sys/class/product_id/
Android# ls
```

Verify "model_name", "part_number", serial_number" and "zone2_data" directories are displayed.

```
Android# cat model_name
```

Verify the model name applicable to the DUT is displayed and is correct.

```
Android# cat part_number
```

Verify the part number applicable to the DUT is displayed and is correct.

```
Android# cat serial_number
```

Verify the serial number applicable to the DUT is displayed and is correct.

```
Android# cat zone2_data. (This step is not applicable to the DM3730/AM3703 Torpedo + Wireless SOM)
```

Verify the NAND entry applicable to the DUT is displayed and correct.

Verify the SDRAM entry applicable to the DUT is displayed and correct.

Guide to interpreting zone2_data:
nor: XXYY nand: ZZAA sdram: BBCC
processor: 0000 platform_bits: 00000001 hardware_revision 00000001

where:

XX is the size of NOR0 flash (power of two in hex)
YY is the size of NOR1 flash (power of two in hex)
ZZ is the size of NAND0 flash (power of two in hex)
AA is the size of NAND1 flash (power of two in hex)
BB is the size of DRAM0 (power of two in hex)
CC is the size of DRAM1 (power of two in hex)

If a field is 0x00 or 0xff, then none populated.

EXAMPLE:
for the 1015061 zone2_data looks like:

nor: ffff nand: 1dff sdram: 1cff
processor: 0000 platform_bits: 00000001 hardware_revision 00000001

In this case, no NOR, NAND0=536870912, no NAND1, DRAM0=268435456, no DRAM1.

16.03 Android Keys

16.03.01 Verify Home/Menu/Back/Power Buttons [GEN-01-007]

On the DUT dev kit, identify the Home, Menu, Back, Power buttons.

Press the “Menu” button

Verify that a menu appears.

Press “Settings.”

Press “Sound.”

Press the “Back” button.

Verify that the Settings options displayed (i.e., the software went one level back.)

Press “Wireless & Networks.”

Press the “Home” button.

Verify that the Home screen is displayed.

Press and hold the “Power” button.

Verify that the “Phone Options” screen is displayed.

Connect Wattson to at least 1 DUT.

Press “Power Off”

Press “OK” to shut down the device.

Verify the DUT is powered off.

*Verify that the power consumption of the SOM is minimal. (*DM3730/AM3703 Torpedo SOM and DM3730/AM3703 Torpedo + Wireless SOM: ~ 0mW. DM3730/AM3703 SOM-LV: ~2mW.*)*

Press the “Power” button again.

Verify the DUT successfully boots up to the home screen.

16.03.02 Reset Button [TBD]

POR

Press the “Reset” button

Verify the DUT boots up to Android successfully

16.03.03 Virtual buttons

POR

For each from (screen in portrait mode, screen in landscape mode):

Put the screen in the proper orientation (see test 01.01.08)

Go to the sound control screen:

Settings > Applications > Settings > Sound

Touch the virtual “home” button on the right edge of the screen in the black space (looks like a little house):

Verify that Android returns to the ‘home’ screen.

Go to the sound control screen again:

Settings > Applications > Settings > Sound

Touch the 'back' virtual button twice (looks like a curved arrow):

Verify that Android is at the Applications screen.

Touch the back virtual button one more time:

Verify that Android is now at the home screen.

Touch the 'recent places' virtual button (looks like overlapping rectangles).

Verify that the settings page shows up on the 'recent places' screen.

16.04 Hardware requirements

16.04.01 The BSP shall support all DM3730/AM3703 Torpedo + Wireless SOM hardware modules as specified in the PRS as of the date of release. [HW-01-001]

Verify that the model numbers listed in the "Model Number Information" section of the DM3730/AM3703 Torpedo + Wireless PRS are ALL currently being tested as a part of this test run.

16.05 User Interface

16.05.01 The BSP shall allow users to put custom data to "/data/app" or "/system/app" directories of Android file system. The custom data in the directories specified above shall be preserved through power cycles. Users shall be able to run custom applications without setting up the desktop development environment. [UI-01-002]

- Copy an application onto an SD card for use in this test. You may copy it onto the boot card if booting from SD card. You may use the Vase.apk application from section 1, or any available Android application. The instructions below will use Vase.apk as an example.

Boot the board with the SD boot card + application, or boot with an SD card + application via NAND boot

Check the 'menu' screen on the Android GUI and make sure that the application is not on the device (not available in the list of useable applications)

If the application is already there, remove it:

Menu > Settings > Applications > Manage applications → select and remove / uninstall

Copy the application from the SD card to /data/app

```
Android# cat /mnt/sdcard/Vase.apk > /data/app/Vase.apk
```

Check the 'menu' screen on the Android GUI and make sure that the application is on the device (available in the list of useable applications)

POR

Verify that the application is still in the /data/app directory.

```
Android# ls /data/app
```

Run the application

Verify that the application was on the 'Menu' screen and that it ran properly.

Remove the application:

Menu > Settings > Applications > Manage applications → select and remove / uninstall

Check the 'menu' screen on the Android GUI and make sure that the application is not on the device (not available in the list of useable applications)

*NOTE: The rest of this test applies to SD booted boards only. NAND booted boards will have the /system/app folder as read only.

Copy the application from the SD card to /system/app

```
Android# cat /mnt/sdcard/Vase.apk > /system/app/Vase.apk
```

Check the 'menu' screen on the Android GUI and make sure that the application is on the device (available in the list of useable applications)

POR

Verify that the application is still in the /system/app directory

```
Android# ls /system/app
```

Run the application

Verify that the application was on the 'Menu' screen and that it ran properly.

Remove the application:

```
Android# rm /system/app/Vase.apk
```

Check the 'menu' screen on the Android GUI and make sure that the application is not on the device (not available in the list of useable applications)

16.05.02 Booting between Android and different OS [UI-01-001]

NOTE: Perform the following tests on one board of each family (one DM3730/AM3703 Torpedo + Wireless SOM, one DM3730/AM3703 Torpedo SOM, one DM3730/AM3703 SOM-LV, etc.)

Insert the Android SD boot card with the SUT on it.

POR

Break into u-boot and erase NAND:

```
OMAP Logic # nand erase.chip
```

POR

Verify the DUT successfully boots up to the Android home screen.

After the DUT boots up to Android, turn off power to the baseboard.

Remove latest Android release SD boot card and insert the Android SD boot card containing the image under test.

Turn on the power to the baseboard

Verify the DUT successfully boots up to the Android home screen. (This makes sure the latest release can be upgraded to the image being tested.)

Turn off power to the baseboard. Insert a Linux SD boot card with the latest image from the download page and turn on power to the baseboard.

Verify the DUT successfully boots up to the Linux login prompt. (This makes sure Android does not inadvertently save the environment to NAND during boot up sequence preventing Linux from booting up.)

After the DUT boots up in Linux, turn off power to the baseboard.

Remove Linux SD boot card and insert the Android SD boot card containing the image under test.

Turn on the power to the baseboard

Verify the DUT successfully boots up to the Android home screen.

16.06 User Guide

16.06.01

Review and perform all the tasks included in the DM37x Android BSP User Guide.

NOTE: Run this test with a "pristine" Linux machine with Ubuntu 10.04 installed.

Verify all function as stated in the User Guide.

16.07 Battery

***NOTE* Make sure that there is a battery connected to the board to test.**

16.07.01 Battery gas gauge

Go to the Battery section of the settings:

Menu > Settings > Battery

Verify that the Battery usage data is listed as available and that you can examine it.

16.08 Release version

16.08.01 Release version

POR

Do a run-time retrieval of the Logic PD Android release version:

```
Android# rm /system/app/Vase.apk
```

Verify that the proper value is returned for the Logic Android release version.

Suspend the SOM (S2)

Wait 5 seconds and resume the SOM (S2)

Check the release version again.

```
Android# getprop logicpd.android.bsp
```

Verify that the proper value is returned for the Logic PD Android release version.

Test 17 - GPS Tests

Purpose

This test verifies that the functional requirements for GPS interface are met.

Setup

As preparation for this test, execute the following in sequence:

- Load any standard image
- Attach the GPS antenna to the SOM and the baseboard.
- Modify the /rootfs/init.rc in the bootable SD card to uncomment the following 3 lines:

#service navl_server /system/bin/navd --android_log NAVD -p3 -nav\"-c/system/etc/gps/config/pathconfigfile.txt\"

#user root

#oneshot

Test Steps

Perform the following subtests and record results in the action log:

17.01 Running GPS Application

17.01.01 GPS_Averaging Application [GPS-01-001] [GPS-01-002]

NOTE: make sure the init.rc file has been modified as described in the Setup section.

If the GPS_Averaging.apk application is not yet installed, install it.

Make sure the GPS_Averaging,apk application is on the SD card

```
DM-37x# cat /mnt/sdcard/GPS_Application.apk > /system/app/.
```

Run the GPS_Averaging application

Verify that the coordinates of the MPLS office are N44° and W93°. (It might take up to 5-10 minutes for the coordinates to become available.)

17.02 GPS Support After Suspend/Resume

17.02.01 Suspend/Resume [GPS-01-006]

NOTE: make sure the init.rc file has been modified as described in the Setup section.

If the GPS_Averaging.apk application is not yet installed, install it.

Make sure the GPS_Averaging,apk application is on the SD card

```
DM-37x# cat /mnt/sdcard/GPS_Application.apk > /system/app/.
```

Turn off Stay Awake so that the DUT can suspend:

Settings > Applications > Development > Stay Awake

Suspend the board (hit S2)

Resume the board (hit S2 again)

Run the GPS_Averaging application

Verify that the coordinates of the MPLS office are N44° and W93°. (It might take up to 5-10 minutes for the coordinates to become available.)

Test 18 – FCC Bluetooth bts file and Wireless ini files

Purpose

This test verifies that the Bluetooth .bts file and the Wifi .ini files approved by FCC are included in the Android BSP.

Setup

Test Steps

Perform the following subtests and record results in the action log.

Perform the following tests on at least one DM3730/AM3703 Torpedo + Wireless SOM.

NOTE:

The FCC approved Bluetooth BTS file is *WL128x_BT_Service_Pack_2.4_LogicPD_v3.bts*.

The FCC approved Wireless INI file is *D7002_v1.0.c_logicpd_v9.ini*.

They are SVN archived in the zip file *Hardware/SOM/DM3730-TOR-WIFI/Docs/FCC_firmware_files.zip* revision 30252 http://viewvc.logicpd.com/viewvc.cgi/Hardware/SOM/DM3730-TOR-WIFI/Docs/FCC_firmware_files.zip?revision=30252&root=eps_svn).

18.01 - Bluetooth

18.01.01 - Verify Bluetooth bts file

NOTE: Perform this test on at least one DM3730/AM3703 Torpedo + Wireless SOM

- Insert a SD card into the Workstation (you can use the SD boot card)
- Using the link to SVN above, down load the file and unzip it onto the card.
- Insert the SD card into the baseboard.
- POR

- Compare the **Logic_TIInit_10.6.15.bts** file on the SOM to the **WL128x_BT_Service_Pack_2.4_LogicPD_v3.bts** file on the sdcard in the FCC_firmware_files/BT folder:

```
For -30  Torpedo + Wireless:

 Android# cmp /system/etc/firmware/Logic_TIInit_10.6.15.bts
/mnt/sdcard/FCC_firmware_files/BT/WL128x_BT_Service_Pack_2.4_LogicPD_v3.bts

For -31  Torpedo + Wireless:

 Android# cmp /system/etc/firmware/Logic_TIInit_10.6.15.bts
/mnt/sdcard/FCC_firmware_files/BT/WL128x_BT_Service_Pack_2.4_LogicPD_v6.bts
```


- *Verify that the 'cmp' command returns nothing.*

18.02 - Wireless

18.02.01 - Verify Logic PD MAC address

- *Verify all DM3730/AM3703 Torpedo + Wireless SOMs have Logic PD MAC address (verified in test 12.01.01)*

18.02.02 - Verify NVS file

NOTE: Perform this test on at least one DM3730/AM3703 Torpedo + Wireless SOM

- Create a SD boot card with the latest Linux BSP release from the Logic PD website.
- Boot Linux
- Copy the wl128x-nvs.bin file onto an SD card.

```
DM-37x#: cp /lib/firmware/ti-connectivity/wl128x-nvs.bin /mnt/mmcblk0p1/.
```

*Note: If you are booting via SD card, you will need to copy the file from Linux onto the SD boot card. If you boot via NAND, any SD card will work.

- Reboot the SOM with the Android SUT

- Compare the .nvs file from Linux with the .nvs file from Android

```
Android# cmp /system/etc/firmware/ti-connectivity/wl128x-nvs.bin /mnt/sdcard/wl128x-nvs.bin
```

- *Verify that the two files are identical (the 'cmp' command returns nothing*

Test 21 - Performance

Purpose

This test verifies that the non-functional requirements for performance are met. Included are performance, power and documentation requirements.

Setup

Test Steps

Perform the following subtests and record results in the action log.

Perform the following tests on one board of each family (one DM3730/AM3703 Torpedo + Wireless SOM, one DM3730/AM3703 Torpedo SOM, one DM3730/AM3703 SOM-LV, etc.)

21.02 - Boot time

21.02.01 - Cold boot to desktop time [PERF-01-002]

This test is to verify that the boot time of the kernel is less than one (1) minute. You will need a stopwatch (or some other way to track time) for this test.

- Boot the SUT using a SD boot card (use SD card included in the kit):
  - Power on the board and start the stopwatch.
  - Stop the stopwatch when the Android main screen appears.
  - *Verify that the boot time for Android was less than one (1) minute.*

  Enter the load time on the performance sheet of the TPL

- Boot the SUT using a NAND boot:
  - Power on the board and start the stopwatch.
  - Stop the stopwatch when the Android main screen appears.
  - *Verify that the boot time for Android was less than one (1) minute.*

  Enter the load time on the performance sheet of the TPL

21.02.02 - Cold boot to splash screen time [PERF-01-001]

This test is to verify that time from cold boot to splash screen is less than 2 seconds. You will need a stopwatch (or some other way to track time) for this test.

- Power on the board and start the stopwatch.
- Stop the stopwatch when the Logic PD splash screen appears.
- *Verify that the time is less than or equal to 2 seconds.*

Enter the load time on the performance sheet of the TPL

21.02.03 - Run and Suspend time [PERF-01-003] [PERF-01-004]

This test is to verify the Run-to-Suspend time and the Suspend-to-Run time. You will need a stopwatch (or some other way to track time) for this test.

- Power on the board
- Turn off Stay Awake so that the DUT can suspend: **Settings > Applications > Development > Stay Awake**

- Hit S2 to enter suspend state and start the stopwatch.
- Stop the stopwatch when the user interface is turned off.
- *Verify that the time is within 1 sec.*

Enter the load time on the performance sheet of the TPL

- Hold S2 to return to the run state and start the stopwatch.
- Stop the stopwatch when the user interface is turned on.
- *Verify that the time is within 1 sec.*

Enter the load time on the performance sheet of the TPL

- Hit S2 to enter suspend state and start the stopwatch.
- Stop the stopwatch when the suspend state is finalized.
- *Verify that the time is within 1.5 sec.*

Enter the load time on the performance sheet of the TPL

- Hold S2 to return to the run state and start the stopwatch.
- Stop the stopwatch when the resume process is complete.
- *Verify that the time is within 3.5 sec.*

Enter the load time on the performance sheet of the TPL

21.03 - Power Usage

21.03.01 Power Usage, Standard demo image [PERF-01-005]

NOTE: THE JUMPER JP6 ON THE SDK2 BASEBOARD *MUST* BE IN THE 2-3 DGND POSITION AND NOT THE 1-2 5V POSITION WHILE USING WATTSON OR THE READINGS WILL BE INCORRECT

- Load the Android image under test.
- Using a USB A to USB mini-b cable, connect the SUT (at the PWR MEAS port) to the workstation.
- Start the Wattson Power Measurement Tool on the workstation
- Using Wattson on the workstation, determine the power consumption of the SOM and SUT while running the standard demo image. Do this by logging the power consumption for 10 seconds and taking the average.
- *Verify that the power consumption of the SOM during Idle state is constant. (Noise is exceptable.)*

Enter the average power consumption on the performance sheet of the TPL

- Turn off Stay Awake so that the DUT can suspend: **Settings > Applications > Development > Stay Awake**
- Hit S2 to enter suspend state
- *Verify that the power consumption (average) of the SOM in Suspend state is less than 10mW.*

Enter the average power consumption on the performance sheet of the TPL

- *Verify that the power consumption (average) of the SUT in Suspend state is less than 30mW.*

Enter the average power consumption on the performance sheet of the TPL

21.04 - Read / Write Performance

21.04.01 SD Read/Write performance

- Load the Android image under test.
- Insert a SD card with at least 10MB of free space.
- Create file of random data (10MB)

```
DM-37x# dd if=/dev/urandom of=/testdata bs=1000 count=10000
```


- Make sure that the file testdata was created

```
DM-37x# ls –l
```

- Calculate the time to copy the file from RAM to SD card

```
DM-37x# time cat /testdata > /mnt/sdcard/testdata; sync
```

- Make sure the file was written to the SD card

```
DM-37x# ls –l /mnt/sdcard
```

Enter the SD write performance on the performance sheet of the TPL

- Erase the testdata file from RAM

```
DM-37x# rm /testdata
```

- Calculate the time to copy the file from SD card to RAM

```
DM-37x# time cat /mnt/sdcard/testdata > /testdata; sync
```

Enter the SD read performance on the performance sheet of the TPL

- Erase the testdata file from SD card

```
DM-37x# rm /mnt/sdcard/testdata
DM-37x# rm /testdata
```

21.04.02 NAND Read/Write performance

- Load the Android image under test.
- Create the NAND directory.

```
DM-37x# mkdir -p /mnt/mtd-nand
```

- View partitions

```
DM-37x# cat /proc/mtd
```

Note: Be sure the following command uses a NAND partition. Use a mtdblock other than 4 if necessary

- Mount the NAND partition.

```
DM-37x# mount -t yaffs2 /dev/block/mtdblock5 /mnt/mtd-nand
```

- Create file of random data (10 MB)

```
DM-37x# dd if=/dev/urandom of=/testdata bs=1000 count=10000
```

- Make sure that the file testdata was created

```
DM-37x# ls –l
```

- Calculate the time to copy the file from RAM to the NAND partition

```
DM-37x# time cat /testdata > /mnt/mtd-nand/testdata; sync
```

- Make sure the file was written to the NAND partition

```
DM-37x# ls –l /mnt/mtd-nand
```

Enter the NAND write performance on the performance sheet of the TPL

- Erase the testdata file from RAM

```
DM-37x# rm /testdata
```

- Calculate the time to copy the file from NAND partition to RAM

```
DM-37x# umount /mnt/mtd-nand
DM-37x# mount –o sync –t yaffs2 /dev/block/mtdblock5 /mnt/mtd-nand
DM-37x# time cat /mnt/mtd-nand/testdata > /testdata; sync
```

Enter the NAND read performance on the performance sheet of the TPL

- Erase the testdata file from the NAND partition

```
DM-37x# rm /mnt/mtd-nand/testdata
DM-37x# rm /testdata
```


21.04.03 RAM Read/Write performance

- Load the Android image under test.
- Check available memory **(df , look at / directory)**. Adjust the count of the dd command below if there is not enough free memory to accommodate 2 copies of the random data file (2 x 10 megs).

  DM-37x# **df**
- Create file of random data (10 MB)

  DM-37x# **dd if=/dev/urandom of=/testdata bs=1000 count=10000**
- Make sure that the file testdata was created

  DM-37x# **ls –l**
- Calculate the time to copy the file from RAM to RAM card

  DM-37x# **time cat /testdata > /testdata2; sync**
- Make sure the file was written to RAM

  DM-37x# **ls –l**

Enter the RAM write performance on the performance sheet of the TPL

- Erase the testdata file from RAM

  DM-37x# **rm /testdata**
- Erase the testdata2 file from RAM

  DM-37x# **rm /testdata2**

21.05 - Power during transmission

21.05.02 Ethernet

POR

netcfg

Verify an IP address was obtained

- Plug USB standard A to USB Mini B cord into Workstation and PWR MEAS port
- Start Wattson on the Workstation
- Using Wattson, measure the average power consumption while running iperf by hitting the “Reset Min/Max/Avg” button on Wattson as you start netperf (see below)
- Connect the Workstation and the SUT to the router via Ethernet cables
- Put the SUT into 10BASE-T mode:

  DM-37x# **netcfg eth0 up**

  DM-37x# **ethtool -s eth0 speed 10 duplex full autoneg on**
- Make sure that the DUT is at: *speed = 10Mb/s*

  DM-37x# **ethtool eth0**
- Get an IP Address for eth0

  DM-37x# netcfg eth0 dhcp

  DM-37x# netcfg
- Run netperf with the SUT in transmit mode

  Android # **mkdir /tmp**

  Android # **netserver**

  Android # **netperf –H <Linux box ip address>**

When netperf begins, hit the reset button on Wattson. Do an Alt+Print Screen before the netperf stops (about 8 seconds) to copy the results from Wattson, then paste them into Paint and make a cut/copy of just the "Monitor" results

Enter the power results on the performance sheet of the TPL

- Put the SUT into 100BASE-T mode:

  DM-37x# **ethtool -s eth0 speed 100 duplex full autoneg on**

- Make sure that the DUT is at: *speed = 100Mb/s*

  DM-37x# **ethtool eth0**

- Run netperf with the SUT in transmit mode

  Android # **netperf –H <Linux box ip address>**

When netperf begins, hit the reset button on Wattson. Do an Alt+Print Screen before the netperf stops (about 8 seconds) to copy the results from Wattson, then paste them into Paint and make a cut/copy of just the "Monitor" results

Enter the power results on the performance sheet of the TPL

21.06 Insertion / Removal Time

21.06.01 Insertion and removal times

There are no requirements on insertion and removal times. Test removed.

Appendix A: Test Progress Log (TPL)

The Test Progress Log (TPL) must be appended at the completion of each individual test case. Only five test result qualifiers are permitted. See the section on test requirements and guidelines for additional information For ease of reference, a list of the acceptable test results has been duplicated here.

The TPL is an excel spreadsheet maintained in the version control system.

Pass - Test passes without any indication of failure.

Pass R/L - Test required redlines to the test procedure to maintain a Pass result.

Concern - Test may or may not have failed due to unexpected or undesired behavior, but is not serious enough to merit a Fail rating. A concern will be entered into the issue tracking system.

Fail – All Devices Under Test (DUT) fail and may represent a significant issue.

Fail Partial – At least one DUT fails and may represent a significant issue.

Blocked - Unable to execute test (prevented by some other error, not supported by the model, etc.). Depending on the reason, blocked issues may need to be entered into the issue tracking system.

N/A – Test is not applicable. See comments for reason the test is N/A.

Example Entries

The following are examples of proper Test Progress Log entries. Note that all of the tests that are not recorded as “PASS” in the Progress Log would also be documented with greater detail in the issue tracking system.

Test #	Test Results	Issue #	Test Date	Initials	Execution Time (hour)	Notes
02.01	Pass		5/7/9	DVE	.25	
02.02	Fail	JIRA-1234	5/7/9	DVE	1.5	
02.03	Concern	JIRA-1235	5/7/9	DVE	.5	Could be SRS or SW or HW issue
02.04	Pass		5/7/9	DVE	1.75	
02.05	Blocked		5/7/9	DVE	.25	Hardware unavailable
02.06	Blocked	JIRA-1240	5/7/9	DVE	2.0	Hardware became too hot

Appendix B: NAND Stress Testing

Purpose

This test stresses the NAND file system and verifies the system is intact following load testing.

Setup

As preparation for this test, execute the following in sequence:

Test Steps

Perform the following subtests and record results in the action log:

1) Start a new log on Tera Term to track the results of this test. When the test is finished, put the logs and an overall results file (that you will create) into a folder called Nand _Stress_test_results in the results folder for the SUT.

2) Create an Android DM3730/AM3703 Torpedo + Wireless SD boot card with the SUT if you do not already have one.

3) In the folder that you got this STP from is another folder called “Android_nand_stress_test_files”. View the “Android_nand_readme.txt” file and make the necessary changes to run the nand stress test.

4) Copy the contents of the Android_nand_stress_test_files from the folder that you got this STP from.

5) Create a LogicLoader SD Boot card with the current version of LogicLoader if you do not already have one.

6) Use the LogicLoader boot card to boot LogicLoader and erase all of nand (adjust blocks if needed)

```
losh> erase /dev/nand0 B0 B4096
```

7) Using the Android DM3730/AM3703 Torpedo + Wireless SD boot card you created, boot the DUT.

8) Make sure Settings > Applications > Development > Stay Awake is checked to avoid having the DUT going into suspend state.

9) Verify the SD card is mounted (look in /mnt/sdcard for your files).

```
DM-37x# ls /mnt/sdcard
```

10) Execute the stress test

```
DM-37x# busybox sh /mnt/sdcard/nand-stress-test.sh
```

Note: If you use Windows to copy the file, it uses CRLF as line endings as opposed to LF (at linux expects). To kill the line endings, do:

```
DM-37x# mkdir /tmp
DM-37x# busybox tr -d '\r' < /mnt/sdcard/nand-stress-test.sh > /tmp/nand-stress-test.sh
DM-37x# busybox sh /tmp/nand-stress-test.sh
```

11) Create a text file of the results of the NAND Stress testing that lists all of the boards tested and how far they got in testing. Put this file in the NAND _Stress_test_results folder in the results folder for the SUT.

Appendix C: Using iperf

iperf is a freeware traffic generation tool. It is a client/server application that should run on both peers, and generates UDP and TCP traffic from the client to the server, according to the configuration described in this appendix.

You should run the server before the client.

A.1 TCP iperf Command

TCP Client Command: This command should be run on the peer that generates and transmits the data. ./iperf –c 'Destination IP' –I 'Interval' –p 'TCP Port' –t 'Time in Seconds' For Example: Running a TCP client from the SUT to the PC behind the AP.

```
# iperf -c 192.168.1.10 -i 2 -p 6000 -t 90
```

Client connecting to 192.168.1.10, TCP port 6000 TCP window size: 16.0 kByte (default)

TCP Server Command: This command should be run on the peer that receives the data. ./iperf –s –I 'Interval' –p 'TCP Port' For Example: Running the TCP server to the SUT.

```
# iperf –s –I 2 -p 6000
```

Server listening on TCP port 6000 TCP window size: 85.3 kByte (default)

A.2 UDP iperf Command

UDP Client Command: This command should be run on the peer that generates and transmits the data. ./iperf –c 'Destination IP' –u –b 'bandwidth' –i 'Interval' –p 'UDP Port' –t 'Time in Seconds' For Example: Running UDP 1Mbps traffic from the SUT.

```
# iperf -c 192.168.1.10 –u -b 1M -i 2 -p 6000 -t 90
```

Client connecting to 192.168.1.10, UDP port 6000 Sending 1470 byte datagrams UDP buffer size: 99.0 kByte (default)

UDP Server Command: This command should be run on the peer that receives the data. ./iperf –s –I –u 'Interval' –p 'UDP Port' For Example: Running the UDP server on the SUT.

```
# iperf –s -i 2 -p 6000 -u
```

Server listening on UDP port 6000 Receiving 1470 byte datagrams UDP buffer size: 99.0 kByte (default)

A.3 iperf with QoS Tagging

Tagging should be run only on the client side. The rest of the command is identical to that on the standard iperf client. No server modification is required. ./iperf –c ‘Destination IP’ –S ‘Tag’ –u –b ‘bandwidth’ –i ‘Interval’ –p ‘UDP Port’ –t ‘Time in Seconds’ For Example: Running 100K voice traffic from the SUT to the PC behind the AP.

```
# iperf -c 192.168.1.10 -S 224 –u -b 100K -i 2 -p 6000 -t 90
```

Client connecting to 192.168.1.10, UDP port 6000 Sending 1470 byte datagrams UDP buffer size: 99.0 kByte (default)

Appendix D: QoS Support in Windows

In order to use the QoS on your Windows PC, you should add a key in the Windows registry. Follow the procedure below to do so.

To add a key to the Windows registry:

1 Select Start ⇩ Run.

2 In the window that opens, enter regedit and click OK. The **Registry Editor** window is displayed:

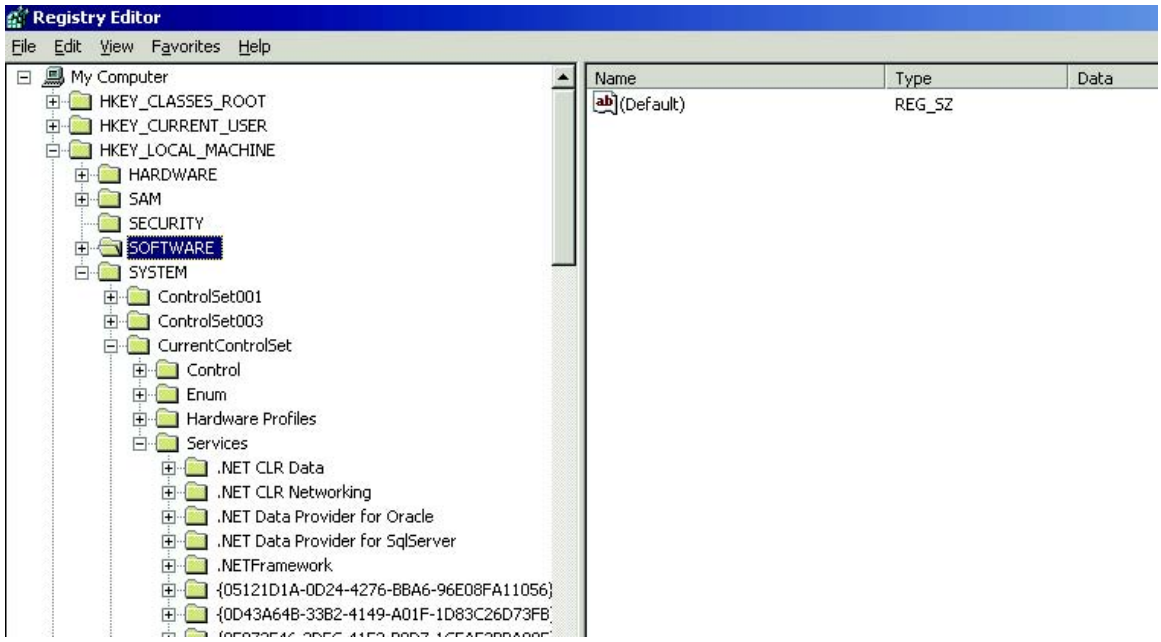


Figure 2: Registry Window

3 Locate the following path in the tree:
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters

4 Right-click in the right pane of the window and select New > DWORD Value, as shown below:

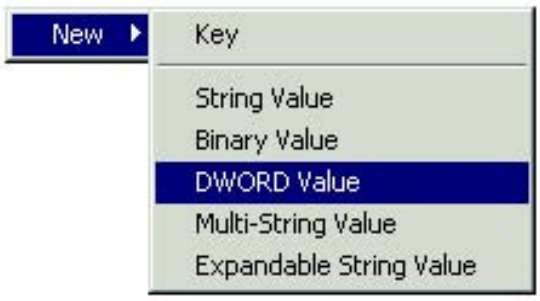


Figure 3: DWORD Right-click Option

The following window is displayed:

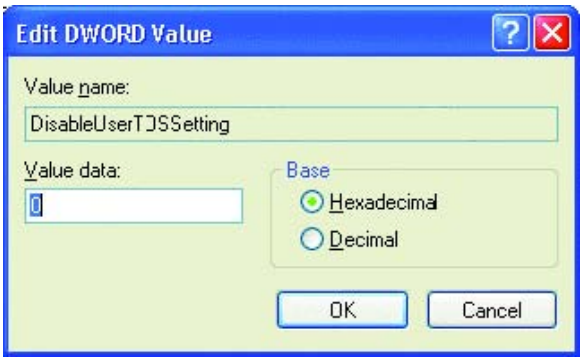


Figure 4: Edit DWORD Value Window

5 In the **Value name** field, enter **DisableUserTOSSetting**.

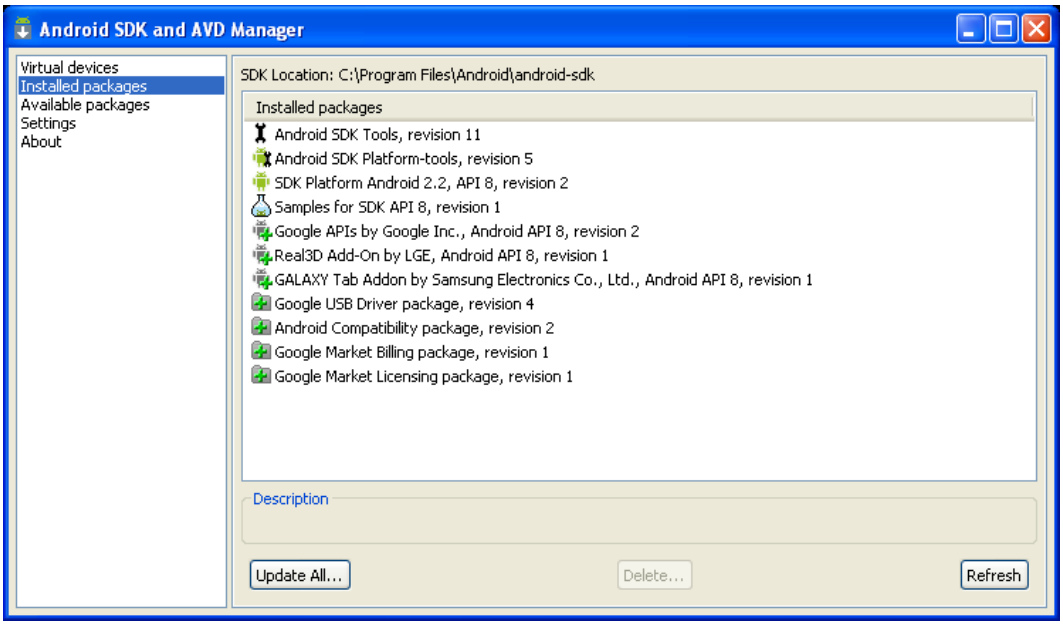
6 In the **Value data** field, enter 0.

7 Restart the PC.

Appendix E: Android Debug Bridge Setup

Android Debug Bridge (ADB)

- Install Java Development Kit (JDK)
- Install Android SDK (http://developer.android.com/sdk/.) If the installer does not detect the JDK, note the workaround documented here: http://stackoverflow.com/questions/4394584/android-sdk-install-problem.
- Add the location of the Android Debug Bridge (ADB)—e.g. in Windows, C:\Program Files\Android\android-sdk\platform-tools— to your path. This can be done under Control Panel / System / Advanced / Environment Variables / System Variables.
- Install the following Android SDK packages:



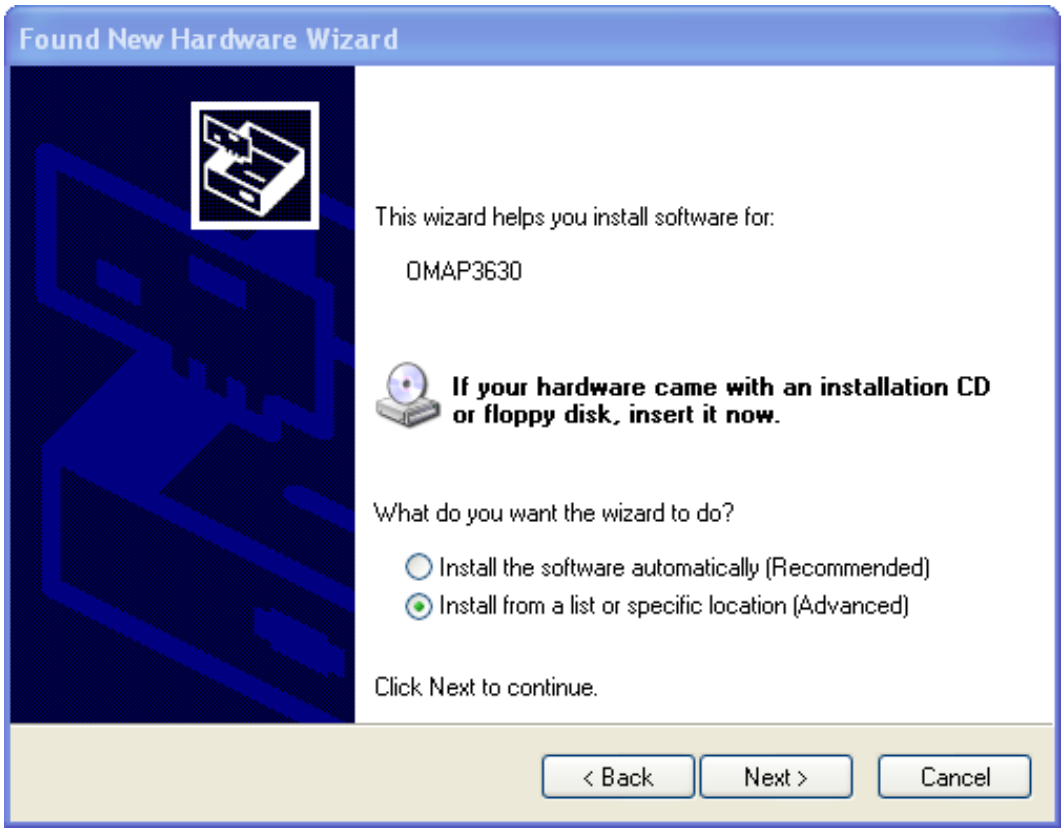
- Add these lines in C:\Program Files\Android\android-sdk\extras\google\usb_driver\android_winusb.inf, under the `[Google.NTx86]` and `[Google.NTamd64]` sections:

```
;DM3730 SOM-LV, DM3730 Torpedo SOM, and DM3730 Torpedo + Wireless SOM
%SingleAdbInterface% = USB_Install, USB\VID_18D1&PID_0001
%CompositeAdbInterface% = USB_Install, USB\VID_18D1&PID_0001&MI_01
```

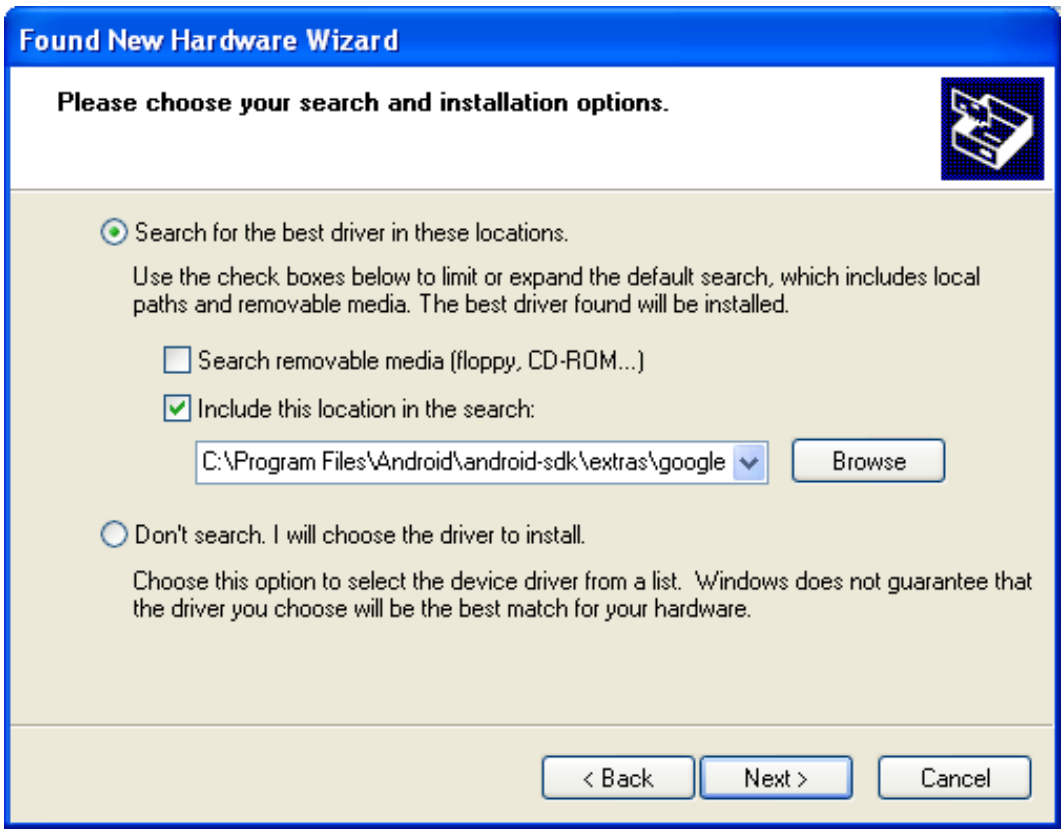
- Add the following line to C:\Documents and Settings\<your user name>\.android\adb_usb.ini

```
18D1
```

- Connect a USB cable from the USB OTG port of a dev kit running Android to your workstation.
- When the hardware is connected, the “Found New Hardware Wizard” will come up. Select “Install from a list or specific location”.



- Select “Include this location in the search” and use the following location: **C:\Program Files\Android\android-sdk\extras\google\usb_driver**



+ Verify ADB is working properly by opening a command prompt and typing

```
adb -d shell
```

If adb is working properly, you should see a # prompt.