ALT1250 Reference Product Installation and Operation Guide

November 2017

6 HaHarash Street, P.O Box 7158, Hod Hasharon 45240, ISRAEL | Tel: +972-74-780-0800 | Fax: +972-9-740-3049

Legal Notice

This document contains proprietary information and is issued for evaluation purposes only. This document may not be photocopied. This document may only be given to those covered by a Non Disclosure Agreement with Altair Semiconductor.

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

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

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

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

Altair products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support life, or for any other application in which the failure of the Altair product could create a situation where personal injury or death may occur. Should Buyer purchase or use Altair products for any such unintended or unauthorized application, Buyer shall indemnify and hold Altair and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, cost, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Altair was negligent regarding the design or manufacture of the part.

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

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

Document Revision Control

Edition	Issued By	Date	Description
Rev 1	Asaf Shabtai	July 2017	Preliminary Release
Rev 2	Asaf Shabtai	September 2017	Modified the 'Device Access over PPP Connection' chapter. Added the 'Booting in Boot-ROM State' section. Added the 'Switching UART Functionality at Run-time' chapter.
Rev 3	Asaf Shabtai	November 2017	Modified 'Embedded Connection Manager (eCM)' chapter – 'ecm' file replaced by 'modem_apps' file Modified the 'FW Debug Logging' chapter Added the 'BSP Files Access' chapter

Table of Contents

List of Figures

List of Tables

1 Introduction

This document provides a step-by-step installation and general operation instructions for working with the ALT1250 Reference Product, based on Altair's ALT1250 chipset.

2 Prerequisites

The Altair's ALT1250 Reference Product is compatible with the following Operating Systems:

- Windows 7 / 8 / 10
- Linux

2.1 HW Components

- Altair ALT1250 based Reference Board (see below)
- BSP files with appropriate calibration values

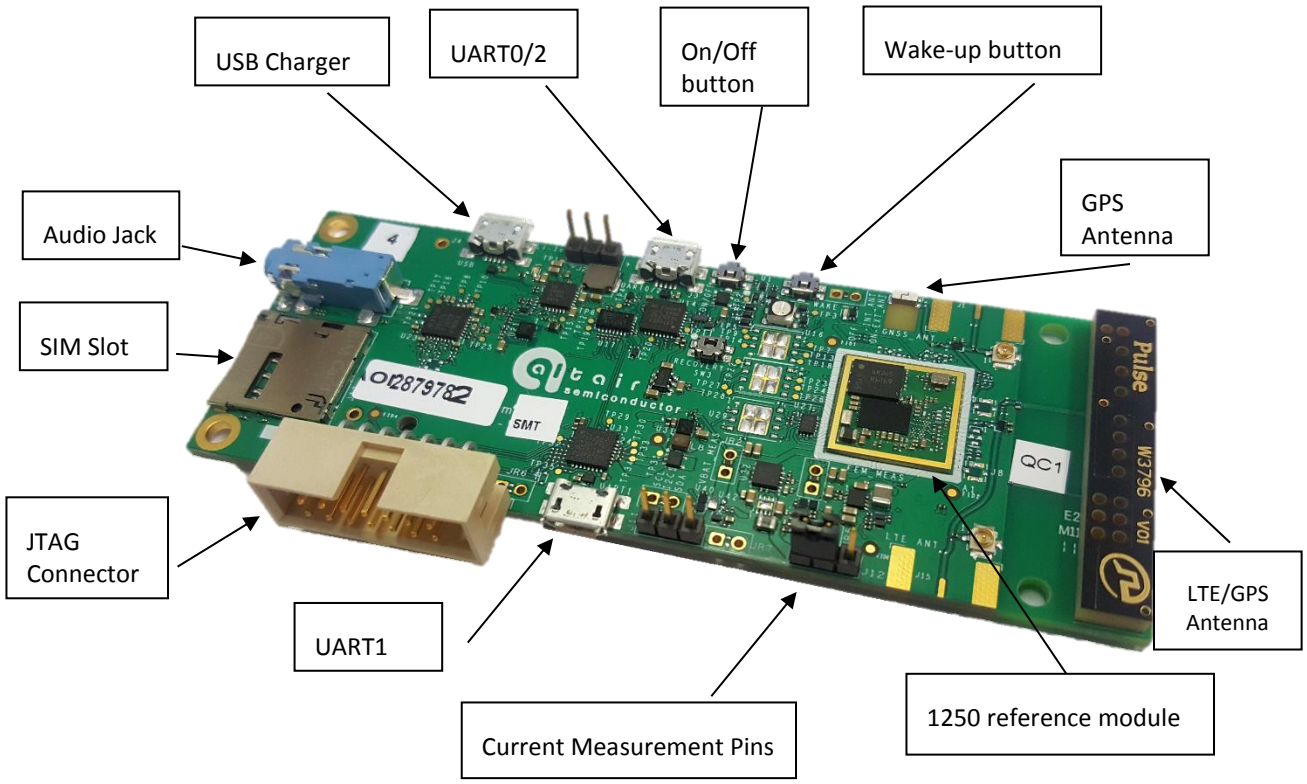
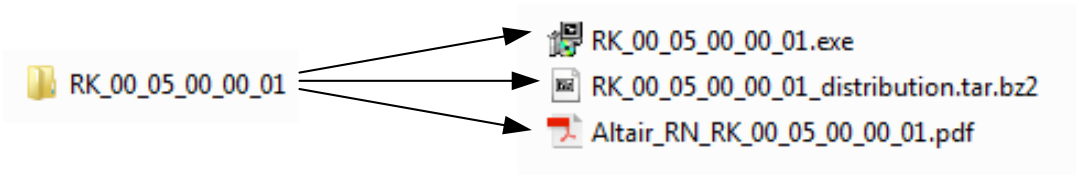


Figure 1. ALT1250 Reference Board

Note: The device must have a serial (NOR) flash with a U-Boot burned on it.

2.2 RiceKrispies (RK) SW Components

- RK_XX_XX_XX_XX_XX.exe - for installation on a Windows machine
- RK_XX_XX_XX_XX_XX_distribution.tar.bz2 - Reference Product distribution kit used for software development on the Altair OpenRTOS platform
- Altair_RN_RK_XX_XX_XX_XX_XX - SW Release Notes



Note: An explanation of the release naming can be found in the Release Notes document.

3 Reference Product Environment

3.1 Installation Packages

⚠ **Attention!** The Reference Product Environment installation is intended only for a Windows PC.

The environment installation is divided to the following separate installation packages:

- RiceKrispies SW – a package containing dedicated SW images per RiceKrispies version release
- PC Tools – a package containing configuration, SW burning, logging and debugging tools
- Production Tools – a package containing a set of tools used for bring-up and production

Note: The RiceKrispies SW package will be provided to customers upon a release of a new RiceKrispies SW version.

The other packages will be provided to the customer from time to time upon issuing new releases internally at Altair. As the other packages are independent, there is no need to upgrade their version every time the RiceKrispies SW version is upgraded (unless stated otherwise by Altair).

3.1.1 RiceKrispies SW Package

Setup File

RK_XX_XX_XX_XX_XX.exe

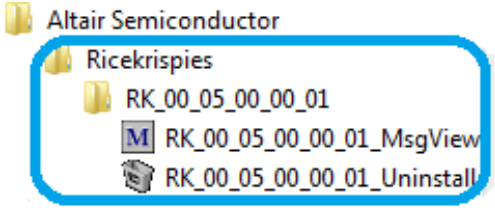
Setup Procedure

This is a simple installation. It requires the user to accept Altair's License Agreement and finish the installation by clicking the 'Close' button.

Installation Result

The following environment will be created under:

"Start->All programs->Altair Semiconductor->Ricekrispies"

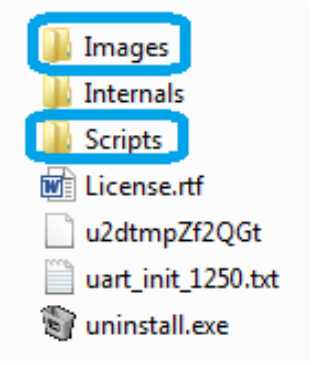


Available Tools

MsgView – application that shows the LTE oriented messages passing between the UE and the eNB.

A dedicated ‘RK_XX_XX_XX_XX_XX’ directory (relevant for the specific RiceKrispies version) will be created under “C:\Program Files (x86)\Altair Semiconductor\Ricekrispies\”.

It will contain the following environment:



The ‘Images’ directory will contain the SW images relevant for the specific RiceKrispies version.

The ‘ImageBurnTool’ application (part of the PC Tools package) uses the images in the ‘Images’ directory and scripts in the ‘Scripts’ directory in order to burn the relevant images to the Reference Product.

3.1.2 PC Tools Package

Setup File

PcTools_XX.XX.XX.XX_setup.exe

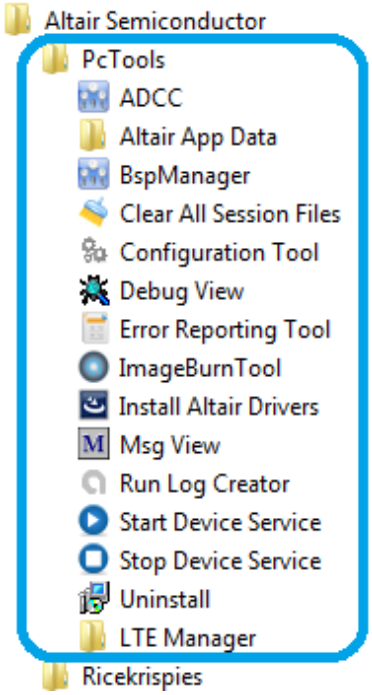
Setup Procedure

This is a simple installation. It requires the user to accept Altair’s License Agreement and follow the installation by clicking the ‘Next’, ‘Install’ and ‘Finish’ buttons when necessary.

Installation Result

The following environment will be created under:

“Start->All programs->Altair Semiconductor->PcTools”



Available Tools

ImageBurnTool – application for burning a new SW version.

Debug View – application for viewing the FW logs generated by the UE.

Msg View – a script that runs the 'MsgView' application of a specific RiceKrispies version, according to the version specified in the 'Product to use' field in Altair's 'Configuration Tool'.

BspManager – application for reading / writing BSP files from / to the UE.

Configuration Tool – application for setting Altair tools related configuration.

Run Log Creator – a script that runs the 'LogCreator' application which parses and streams binary FW logs from the UE side to host PC applications (Debug View / Msg View).

All other tools are intended for Altair's internal use.

Note: At the first installation of the PC Tools, the following installations (required by Altair's PC Tools) will take place as well:

- Python
- MCR (Matlab Component Run-time)
- Wireshark (including WinPcap)

3.1.3 Production Tools Package

Setup File

ProductionTools_XX.XX.XX.XX_setup.exe

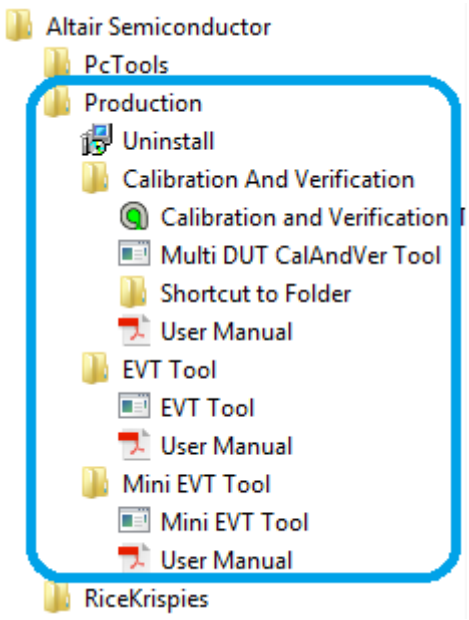
Setup Procedure

This is a simple installation. It requires the user to accept Altair's License Agreement and follow the installation by clicking the 'Next', 'Install' and 'Finish' buttons when necessary.

Installation Result

The following environment will be created under:

"Start->All programs->Altair Semiconductor->Production"



Available Tools

Calibration and Verification – library for production usage (including the 'Calibration and Verification' tool).

EVT Tool – library for bring-up usage (including the 'EVT' tool).

Mini EVT Tool – library for RF testing (including the 'Mini EVT' tool).

4 Software Update

Note: All UART ports references in this chapter are aligned to the UART ports assignment of Altair's ALT1250 Reference Board.

For customer designs which have different UART ports assignment than Altair's Reference Board, although the UART ports to be used are different, the operation and steps to be taken are the same as for Altair's Reference Board.

In case different / additional setting is required for customer designs, it will be stated.

4.1 ImageBurnTool

The ImageBurnTool application is a Windows application used for burning Altair's SW images and configuration files on the Reference Product. It is installed on a Windows PC as part of Altair's PcTools installation.

Before using it, the user should connect the device to a PC (on which ImageBurnTool is installed) via the 'UART0/2' micro-USB connector and power it up. After the first plug-in, the device's necessary Windows drivers will be installed automatically on the PC.

After the necessary PC drivers are installed and the device is ready to be used, the ImageBurnTool should be run from:

"Start->All programs->Altair Semiconductor->PcTools->ImageBurnTool"

For reading the ImageBurnTool's version and for more operational tips, please refer to the 'About' tab of the tool.

4.1.1 General Burning Process

Via the 'Version' field, the user should direct the ImageBurnTool to the relevant version from which the SW images should be taken - using one of the following options:

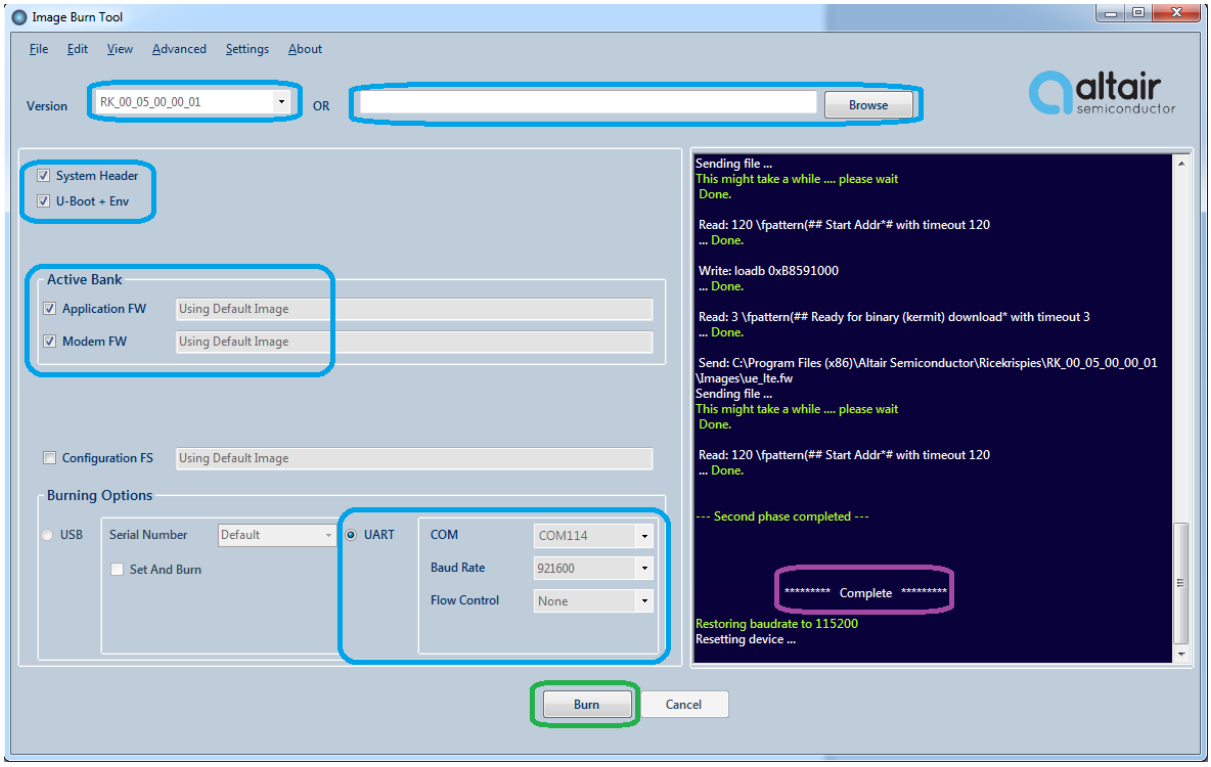
- The 'Version' drop-down list – which lists all the RiceKrispies installations that are under:

  "C:\Program Files (x86)\Altair Semiconductor\Ricekrispies"
- The 'Version' browsing option (on the top right side of the tool) – which allows choosing a specific version folder.

Note: If the user chooses option #2, he needs to make sure that the version folder has the same structure as an official 'RK_XX_XX_XX_XX_XX' version, meaning contains the same sub-folders ('Images', 'Internals' and 'Scripts').

After that, the user should check the boxes according to the SW components he wishes to burn (as explained in the 'Burning Use Cases' section below).

Once all settings have been made, the user should press the 'Burn' button.



The progress of the burning process will be displayed in the log console embedded in the right side of the ImageBurnTool.

The user should expect to see the "*** Complete ***" message (circled in purple in the screenshot above) which indicates the burning process has finished.

Should any errors occur during the burning process, they will appear in the log console and the burning process will halt.

⚠ **Attention!** During burning of the System Header / U-Boot, do not stop the burning process. This may cause the device not to boot after the burning process.

4.1.2 Burning Use Cases

Choosing the SW components to burn as well as the burning procedure itself are determined according to the device's status. There are several burning use cases a user may encounter as shown below:

- Burning a device with an empty serial flash.
- On-going SW update.
- Recovering a device with faulty System Header / U-Boot.

4.1.2.1 Burning a Device with an Empty Serial Flash

When burning a device with an empty serial flash, **all** SW components must be burned ('System Header', 'U-Boot', 'Application FW', 'Modem FW' and 'Configuration FS').

Burning of such device should be done in 'Recovery' mode as explained in the 'Recovery Burning Mode' section below.

4.1.2.2 On-going SW Update

In general, after a device is already burned, the only components that should be burned from time to time are the 'Application FW' and 'Modem FW'.

If the device's U-Boot needs to be burned as well, it will be stated in the SW version's release notes document.

4.1.2.3 Recovering a Faulty Device

A device with a faulty 'System Header' and / or 'U-Boot' cannot boot properly.

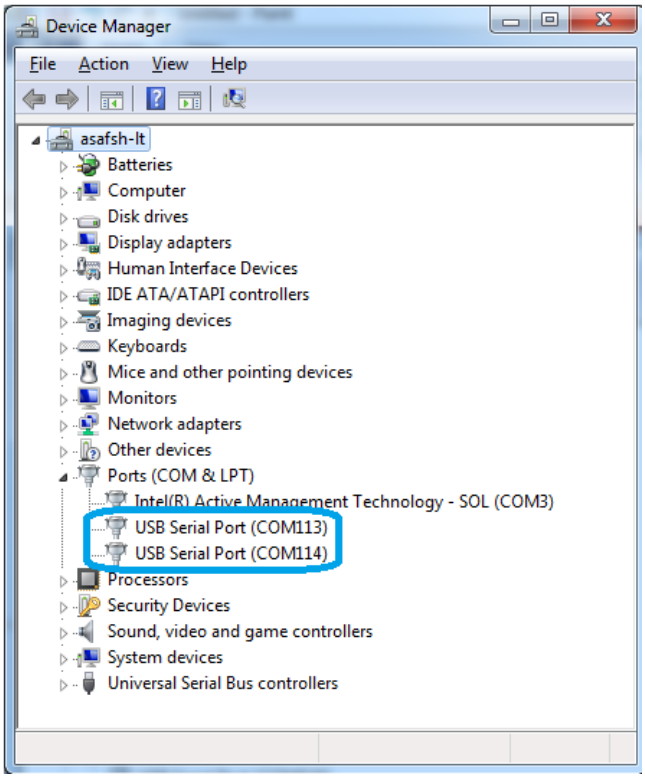
Such device must be recovered, meaning its 'System Header' and 'U-Boot' must be burned in 'Recovery' mode as explained in the 'Recovery Burning Mode' section below.

4.1.3 Burning Modes

The ImageBurnTool provides several burning modes which can be set in the 'Advanced' menu (as explained below).

The burning process is done over a UART connection to the device, therefore the UART serial cable must be connected to the device's 'UART0/2' micro-USB connector prior to burning.

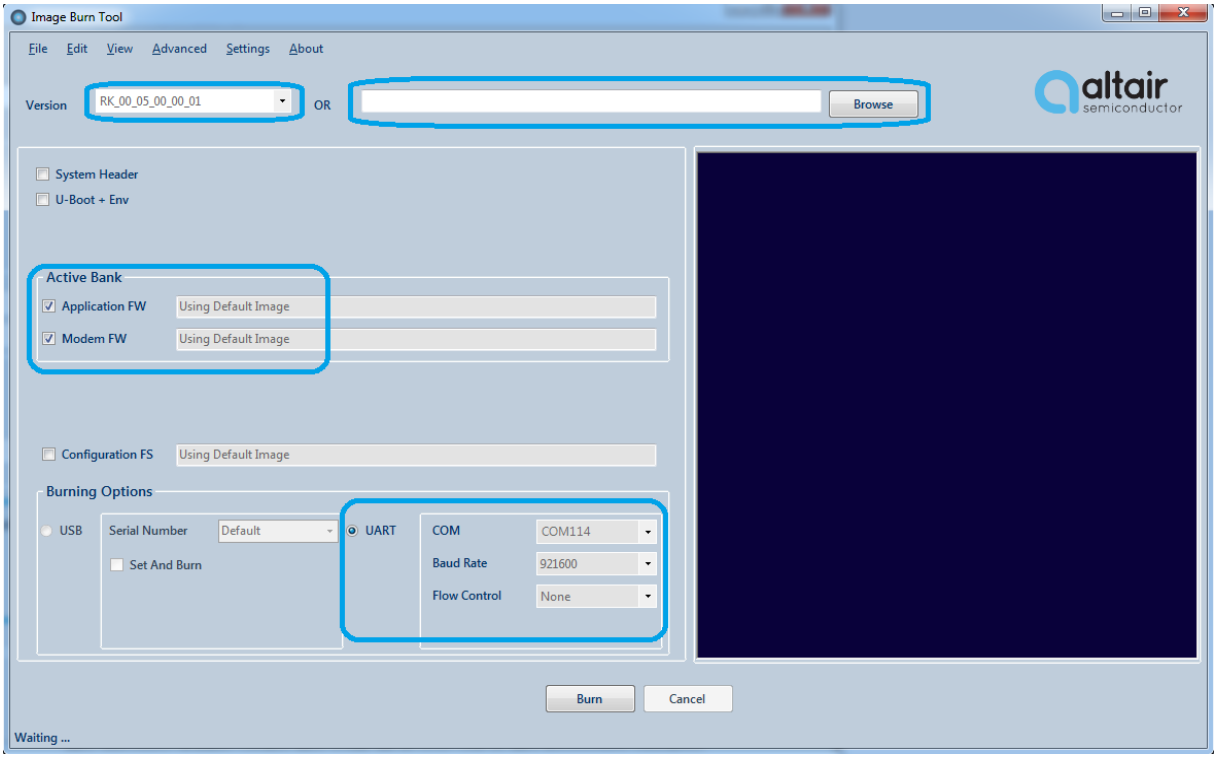
In the PC's Device Manager – locate the 'USB Serial Port' COM ports (there are 2 COM ports) that are opened for the 'UART0/2' interface, as seen in the screenshot below.



The relevant COM port is the one with the greater port number. In the screenshot above it would be COM114.

This COM port will be referred to as the "burning COM port" in the burning steps below.

4.1.3.1 Default Burning Mode



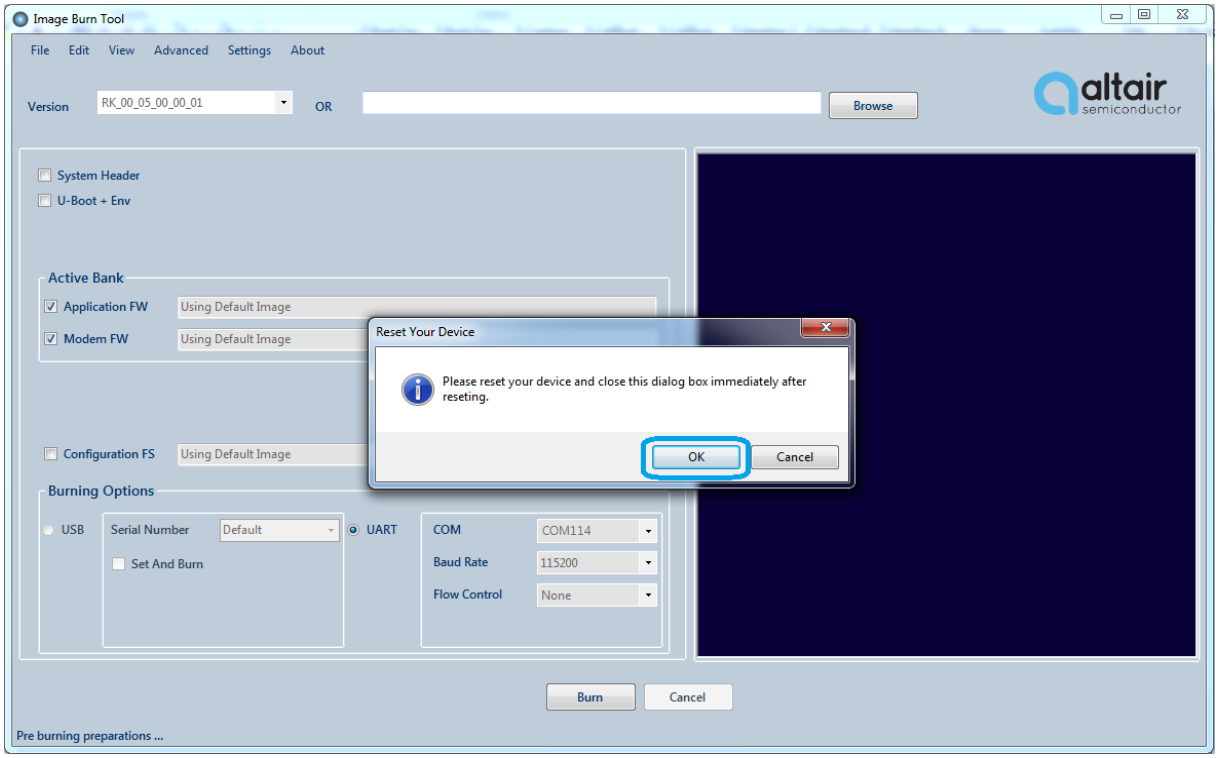
In the 'Default' burning mode, the SW images that will be used for the burning process will be chosen according to what was set in the 'Version' field, as follows:

- In case the 'Version' drop-down list was set, the images will be taken from – "C:\Program Files (x86)\Altair Semiconductor\Ricekrispies\RK_XX_XX_XX_XX_XX\Images", where RK_XX_XX_XX_XX_XX is the version which was chosen in the 'Version' drop-down list.
- In case the 'Version' browsing option was set, the images will be taken from the 'Images' folder under the path that was set in the browsing window.

The burning process should be as follows:

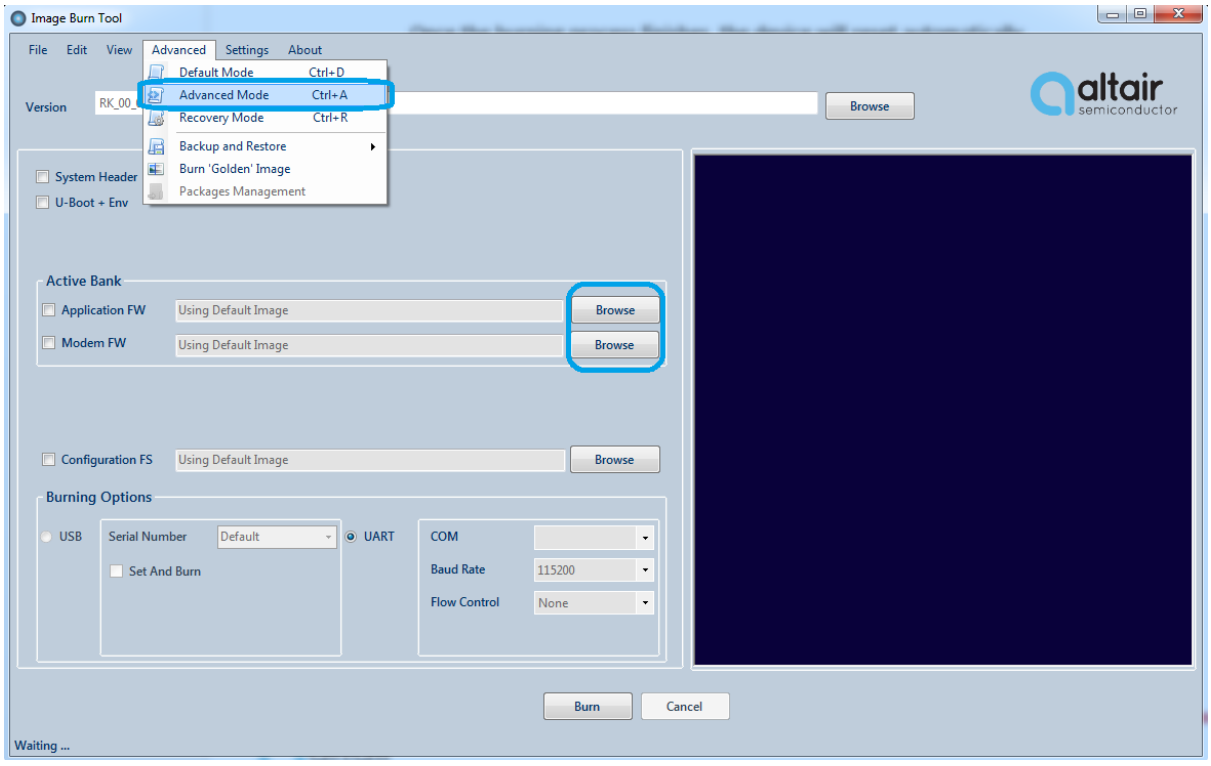
1. Connect the device to the PC.
2. In the ImageBurnTool – choose the SW components you wish to burn as shown above.
3. In the ImageBurnTool – set the UART parameters as follows:
   - 'COM' should be set to the "burning COM port"
   - 'Baud Rate' should be set to '921600'
   - 'Flow Control' should be set to 'None'
4. Start the burning process by pressing the 'Burn' button.

   At this point, a pop-up dialog box will appear and the user will be requested to reset the device and immediately press the 'OK' button in the dialog box (as shown below).



Once the burning process finishes, the device will reset automatically.

4.1.3.2 Advanced Burning Mode



The burning process in the 'Advanced' mode is the same as in the 'Default' mode.

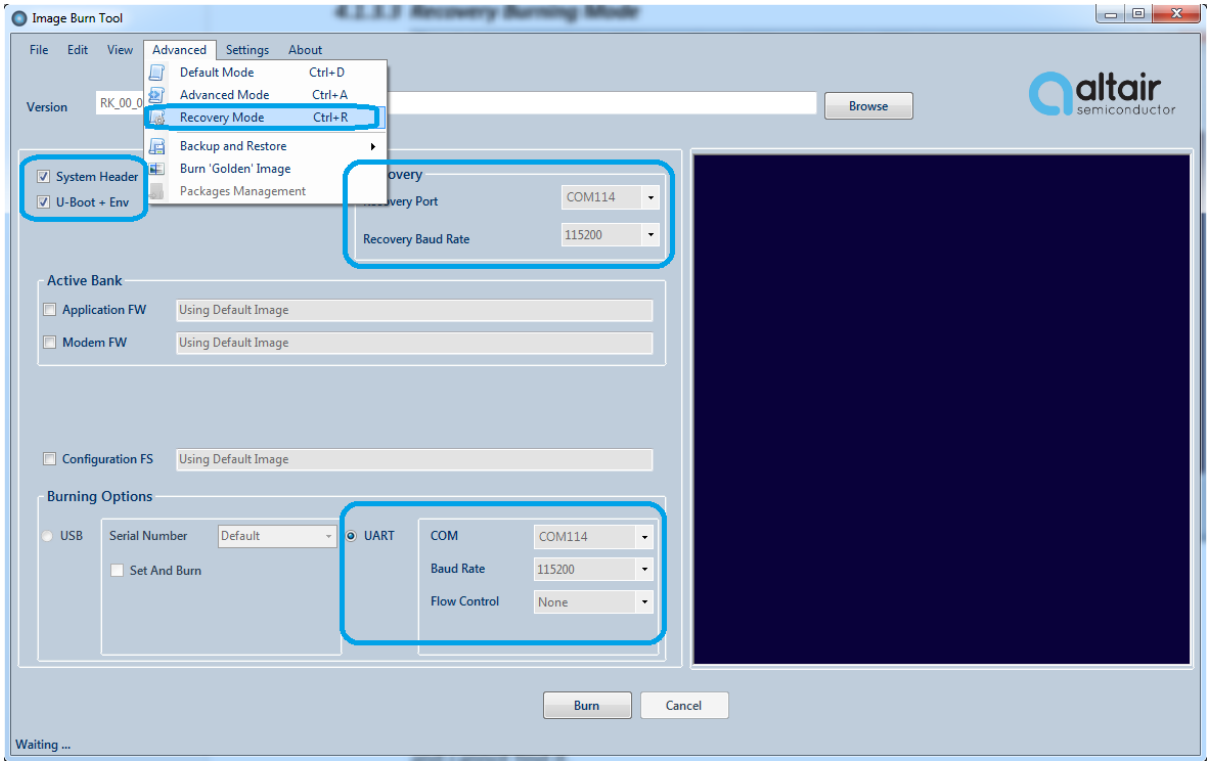
The difference between the 'Advanced' mode and the 'Default' mode is that in the 'Advanced' mode the user can choose to use SW images other than the default ones.

As shown above, a new 'Browse' option is enabled for each SW component.

⚠ **Attention!** In General, each set of 'Application FW' and 'Modem FW' must be of the same SW version.

There are special cases (mainly for debugging purposes) in which one of these components may be taken from a different SW version.

4.1.3.3 *Recovery Burning Mode*



The 'Recovery' burning mode is a special mode intended for:

- Burning a device with an empty serial flash.
- Recovering a device where the 'System Header' and / or 'U-Boot' are faulty.

In order to burn a device in 'Recovery' mode, the device must be in 'Boot-ROM' state. The device enters such state automatically when its serial flash is empty or when the Boot-ROM searches for the System Header on the device's serial flash (at boot time) and cannot find it.

In case the System Header exists in the serial flash and is found by the Boot-ROM, but the U-Boot is faulty or does not exist on the serial flash, the user will need to force the device into Boot-ROM state in order to burn in 'Recovery' mode (how to do it will be explained by Altair if needed).

When the device is in the Boot-ROM state, the tool will load a U-Boot onto the device's RAM via UART, and use it to continue to the burning process.

When burning in 'Recovery' mode, the user should set the following:

- Choose the 'System Header' and 'U-Boot + Env' components.
- 'Recovery Port' is the COM port through which the U-Boot is loaded into the RAM before the components are burned to the flash – it should be set to the "burning COM port".
- 'Recovery Baud Rate' is the rate in which the U-Boot is loaded into the RAM before the components are burned to the flash – the user may choose to set it to be other than the default rate ('115200').
- The UART parameters should be set as follows:
  - 'COM' should be set to the "burning COM port".
  - 'Baud Rate' should be set to '115200'.
  - 'Flow Control' should be set to 'None'

After setting the above, the burning process is done the same as a regular burning, except that the user will not be asked to reset the device prior to the burning.

Note: Before burning, the user must verify that the "burning COM port" is not occupied by other applications.

4.1.4 SW Components Legend

Table 1. SW Components Legend

Component	Related file(s) (under 'Images' directory)	Description
System Header	sysHeader.bin.alt1250	Used by Boot-ROM for pointing to the U-Boot and authenticating it (if security is enabled)
U-Boot + Env	u-boot.bin	U-Boot image for serial flash (including U-Boot environment)
Application FW	AppFW_flash.bin	Image of the Application FW (including the OpenRTOS)
Modem FW	ue_lte.fw	Image of the LTE Modem FW
Configuration FS	ConfigFS.bin	Image for the OpenRTOS BSP and configuration files

Attention! Since the 'Configuration FS' component on the device contains the calibrated BSP files, it **should not** be burned as part of an on-going SW update process.

It should only be burned when burning a device with an empty serial flash or when instructed by Altair.

5 Device Operation via CLI

Note: All UART ports references in this chapter are aligned to the UART ports assignment of Altair's ALT1250 Reference Board.

For customer designs which have different UART ports assignment than Altair's Reference Board, although the UART ports to be used are different, the operation and steps to be taken are the same as for Altair's Reference Board.

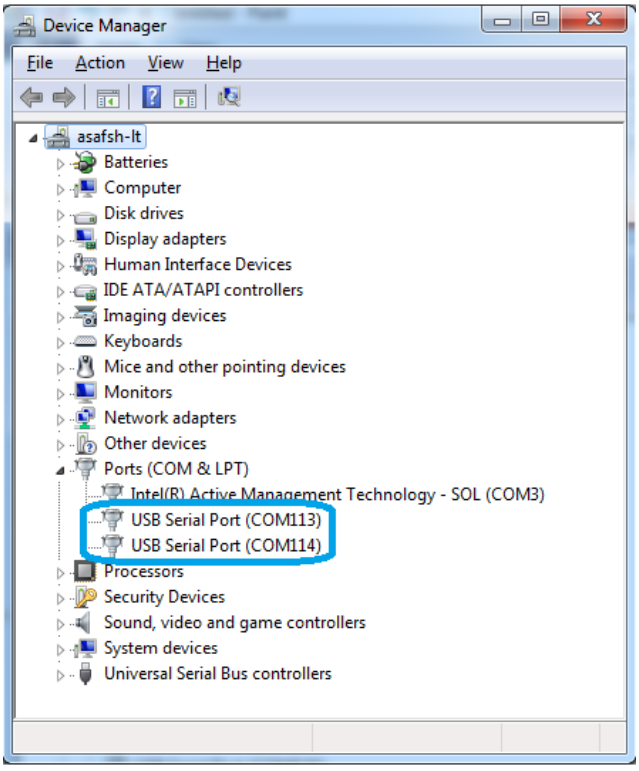
In case different / additional setting is required for customer designs, it will be stated.

The Reference Product supports a CLI (Command Line Interface) mechanism that allows running a set of predefined commands for different uses.

Accessing the CLI can be done via a serial UART connection, by direct serial access or by Telnet access (Telnet access is explained in the 'Device Access over PPP Connection' chapter).

In order to access the CLI via serial UART, the user needs to connect to the device's 'UART0/2' micro-USB connector (same as done in the SW burning process).

In the PC's Device Manager – locate the 'USB Serial Port' COM ports (there are 2 COM ports) that are opened for the 'UART0/2' interface, as seen in the screenshot below.



The relevant COM port is the one with the greater port number. In the screenshot above it would be COM114.

Open a terminal shell (baud rate: 115200, flow control: none) to this COM port.

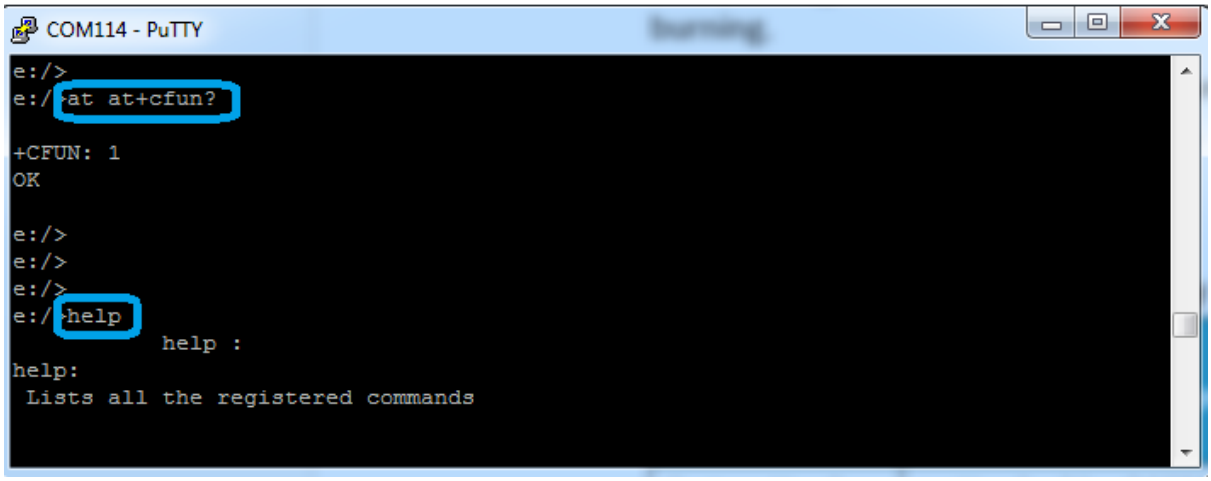
The list of supported commands can be seen when running the 'help' command.

The CLI supports sending AT commands to the modem, as follows:

at <AT command>

For example (as seen in the screenshot below):

at at+cfun?



⚠ **Attention!** When the device is reset while a terminal shell is opened for a specific port, the next time the device will boot, the PC might issue and expose a new COM port instead of the occupied port.

Therefore, it is advised that upon every reset of the device, the user will close all the terminal shells which were opened for the device before resetting the device.

6 Embedded Connection Manager (eCM)

The Embedded Connection Manager (eCM) is a SW based module running on the ALT1250 application processor. It is responsible for establishing, managing and maintaining the LTE connection according to user / operator pre-configurations.

6.1 LTE Connection Mode

The eCM supports 2 methods for establishing the LTE connection:

- Automatic connection – the UE connects automatically to the LTE network after system boot.
- Manual connection – the UE connects to the LTE network only upon user connection initiation.

The LTE connection method is configured in the 'AutoConnectMode' parameter which is part of the 'modem_apps' configuration file.

When 'AutoConnectMode' is set to 'true' – the UE works in automatic connection mode.

When 'AutoConnectMode' is set to 'false' – the UE works in manual connection mode.

6.1.1 Reading the LTE Connection Mode

Reading the current LTE connection mode can be done via CLI, by running:

config -g modem_apps.Mode.AutoConnectMode

6.1.2 Setting the LTE Connection Mode

Setting of the LTE connection mode can be done via CLI, by running:

config -s modem_apps.Mode.AutoConnectMode <true / false>

Examples:

- Setting 'AutoConnectMode' to true –
  config -s modem_apps.Mode.AutoConnectMode true
- Setting 'AutoConnectMode' to false –
  config -s modem_apps.Mode.AutoConnectMode false

After the 'AutoConnectMode' parameter is set, the user needs to run the 'shutdown' CLI command or the 'atz' AT command in order for the setting to be saved in the device's serial flash.

After saving the modified setting, it will get into effect after the next device reboot (the 'shutdown' / 'atz' commands trigger a reboot).

7 FW Debug Logging

Note: All UART ports references in this chapter are aligned to the UART ports assignment of Altair's ALT1250 Reference Board.

For customer designs which have different UART ports assignment than Altair's Reference Board, although the UART ports to be used are different, the operation and steps to be taken are the same as for Altair's Reference Board.

In case different / additional setting is required for customer designs, it will be stated.

The Reference Product supports a FW debug logging mechanism for debugging purposes.

This mechanism provides logs redirection to a host PC (which the device is connected to) for collection and analysis by dedicated PC tools.

In order for the host PC to be able to retrieve the logs, the configuration below is required.

7.1 Setting the Logging Port Parameters in the Device

There are several parameters which characterize the logging channel.

These parameters are part of the U-Boot environment, and should be set as explained below.

For accessing U-Boot environment parameters, please reset the device and open a U-Boot terminal at boot time (as explained in the 'Accessing the U-Boot at Boot Time' section).

1. **'othbootargs' ('phy_uart') parameter**

   Note: This parameter is relevant when working with a design in which the UART ports assignment is different than Altair's.

   It is not relevant when working with an Altair Reference Board.

   By default, the logs are directed from the device to the PC over the 'UART1' interface, whereas the CLI and AT channel / PPP are operational over 'UART0' and 'UART2' respectively.

   This setting is relevant for Altair's Reference Board where there are 2 physical UART connectors: 'UART0/2' and 'UART1' (as can be seen in the Reference Board's picture above).

   However, customer designs in which the UART ports assignment is different than Altair's (only one UART port for both CLI and logging), will require pre-setting of the logging port in order for the logging to be directed properly.

   Verify it is configured correctly by running the following command in the U-Boot terminal:

   printenv othbootargs

If the ‘othbootargs‘ parameter does not exist in the U-Boot environment or its value is not ‘phy_uart=2’, please set it by:

setenv othbootargs phy_uart=2

saveenv

2. **‘mac_log_uartbaudrate’ parameter**

   The logging channel baud rate should be set to ’921600’.

   Verify it is configured correctly by running the following command in the U-Boot terminal:

   printenv mac_log_uartbaudrate

   If the ‘mac_log_uartbaudrate‘ parameter does not exist in the U-Boot environment or its value is not ‘921600’, please set it by:

   setenv mac_log_uartbaudrate 921600

   saveenv

3. **‘mac_log_uartflowctrl’ parameter**

   The logging channel flow control should be enabled.

   Verify it is configured correctly by running the following command in the U-Boot terminal:

   printenv mac_log_uartflowctrl

   If the ‘mac_log_uartflowctrl‘ parameter does not exist in the U-Boot environment or its value is not ‘3’, please set it by:

   setenv mac_log_uartflowctrl 3

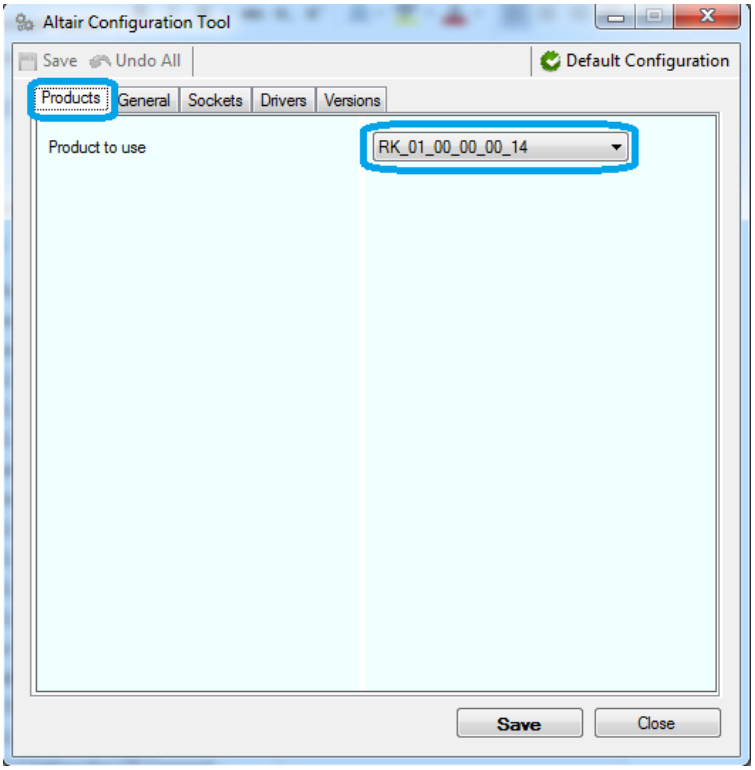
   saveenv

Notes:

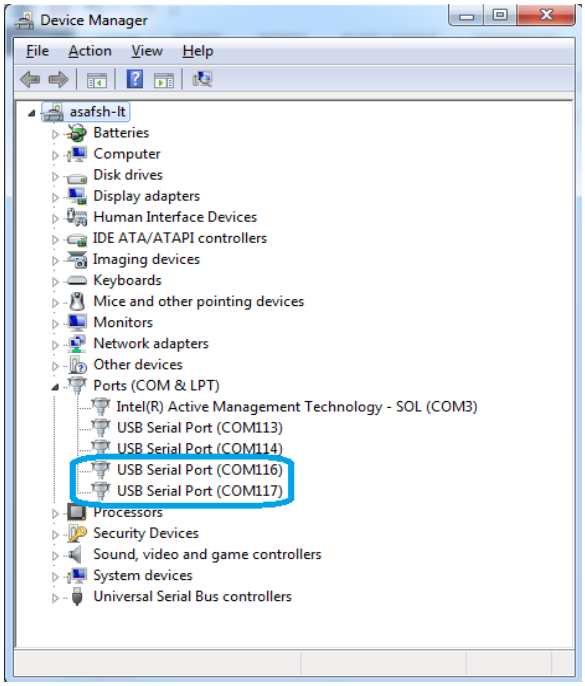
- If one of the parameters above have been modified, the device should be reset in order for the setting to get into effect.
- As mentioned in the ‘U-Boot Upgrade‘ section below, when burning a new U-Boot, all the U-Boot environment parameters are over-written by default values. Therefore, the user must set these parameters again after every U-Boot upgrade.

7.2 Configuration Tool Setting

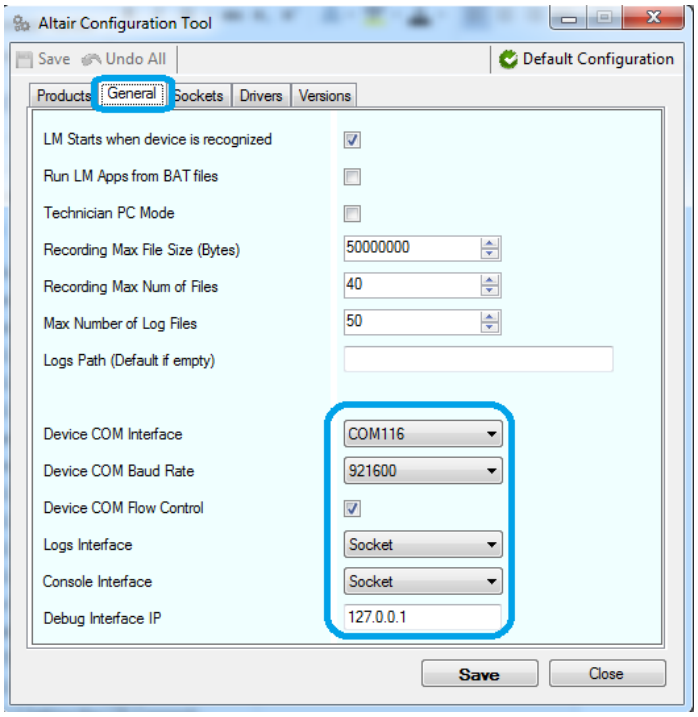
1. Open Altair's 'Configuration Tool' from:
   "Start->All programs->Altair Semiconductor->PcTools"
2. Under the 'Products' tab –
   The user must verify the 'Product to use' parameter is set according to the RiceKrispies SW version that is burned on the device being used.



3. Under the 'General' tab –
   a. The interface through which the logs are transferred to the PC is the device's 'UART1' interface.
      Connect the PC to the device's 'UART1' micro-USB connector.
      In the PC's Device Manager – locate the 'USB Serial Port' COM ports (there are 2 COM ports) that are opened for the 'UART1' interface, as seen in the screenshot below.
      The logging COM port is the one with the lesser port number (in the screenshot below it would be COM116).
      The 'Device COM Interface' parameter should be set to the logging COM port.



b. The ‘Device COM Baud Rate’ should be set to ‘921600’.
c. The ‘Device COM Flow Control’ check-box should be checked.
d. The ‘Logs Interface’ and the ‘Console Interface’ should be set to ‘Socket’.
e. The ‘Debug Interface IP’ should be set to ‘127.0.0.1’.



4. Press ‘Save’ (and approve the saving) in order to save the configuration.
5. Press ‘Close’.

7.3 Viewing the Logs

For viewing the logs, open the 'Debug View' tool from:

"Start->All programs->Altair Semiconductor->PcTools"

⚠ **Attention!** When the device is reset while a terminal shell is opened for a specific port, the next time the device will boot, the PC might issue and expose a new COM port instead of the occupied port.

Therefore, it is advised that upon every reset of the device, the user will close all the terminal shells which were opened for the device before resetting the device.

8 U-Boot Access and Environment Setting

Note: All UART ports references in this chapter are aligned to the UART ports assignment of Altair's ALT1250 Reference Board.

For customer designs which have different UART ports assignment than Altair's Reference Board, although the UART ports to be used are different, the operation and steps to be taken are the same as for Altair's Reference Board.

In case different / additional setting is required for customer designs, it will be stated.

8.1 Accessing the U-Boot at Boot Time

The U-Boot program may be accessed at boot time.

In order to allow such access, the 'bootdelay' parameter (part of the U-Boot environment) must be configured accordingly as explained in the 'bootdelay' section below.

A user can access the U-Boot at boot time by opening a terminal shell (serial connection, baud rate: 115200, flow control: none) to the relevant COM port which is exposed by the U-Boot during the booting, and stopping the U-Boot by pressing 'Enter' during the U-Boot countdown.

8.1.1 'bootdelay'

The 'bootdelay' parameter determines the time period (in seconds) after the U-Boot is up, in which the U-Boot "waits" for possible access by the user. The boot delay must be greater than 0 and allow enough time for user's access.

The boot delay may be configured by running the following commands via a U-Boot shell:

setenv bootdelay T (T = time in seconds)

saveenv

This will take effect after the next device reboot.

The 'bootdelay' parameter's current value may be displayed by running the following command via a U-Boot shell:

printenv bootdelay

8.1.2 Disabling Access to the U-Boot at Boot Time

In order to disable the access to the U-Boot at boot time, the 'bootdelay' value must be set to '0'.

In order to set the 'bootdelay' to '0', user should run the following command via a U-Boot shell:

setenv bootdelay 0

This will take effect after the next device reboot.

⚠ **Attention!** When the 'bootdelay' is set to '0', the access to the U-Boot at boot time is disabled.

Such access is required for upgrading the device via the U-Boot using Altair's ImageBurnTool. Therefore, in case the user wishes to use the ImageBurnTool, the 'bootdelay' parameter must be set to a value greater than '0' (preferably greater than 5).

If the user wishes to upgrade the device while the 'bootdelay' is set to '0', it will have to be done in 'Recovery' mode.

8.2 GPIO Configuration in U-Boot Environment

The U-Boot has the ability to configure a pin as a GPIO and set its direction (input / output) and value (0 / 1) at boot time.

Such configuration may be done by modifying the 'preboot' parameter in the U-Boot environment. This modification includes adding a specific command for each pin according to the pin's intended functionality.

The format of the command is:

gpio <operation> <pin>

The available 'operations' are:

- input – configures the specific pin as a GPIO with direction input and prints its value.
- set – configures the specific pin as a GPIO with direction output and sets its value to '1'.
- clear – configures the specific pin as a GPIO with direction output and sets its value to '0'.
- toggle – configures the specific pin as a GPIO with direction output and sets its value to the value opposite to its value prior to the setting ('0' is set to '1' and vice versa).

The 'pin' number should be provided by Altair for each pin the user wishes to configure, according to Altair's pre-defined mapping rule.

Note: When adding several commands, each command will be separated by a semicolon (as shown in the examples below).

8.2.1 Setting the 'preboot'

Via a U-Boot Shell -

1. Read the current 'preboot' value by running:

   printenv preboot

   Output will be (for example):

   preboot=if test -n $prebootcmd; then echo; echo Running pre-boot command; run prebootcmd;fi;

   Note: As this value may be changed by Altair in the future (relevant after U-Boot is updated), it is important to read the current value of 'preboot'

before modifying it.

2. Set the new ‘preboot’ value by running:

   setenv preboot “NEW VALUE”

   “NEW VALUE” should be the original value, adding the relevant command(s) at the end of it.

   Note: The quotation marks (“ “) of “NEW VALUE” are part of the syntax.

3. The GPIO configuration will take place at the device’s next boot up.

Examples:

- In order to configure pin ‘GPIO6’ (mapped to pin ‘7’) as a GPIO and set its direction to ‘output’ and its value to ‘1’, the user should set the ‘preboot’ as follows:

  setenv preboot "if test -n $prebootcmd; then echo; echo Running pre-boot command; run prebootcmd;fi;**gpio set 7;**"

- In order to configure pin ‘GPIO6’ as mentioned in the previous example and also configure ‘GPIO7’ (mapped to pin ‘8’) as a GPIO and set its direction to ‘input’, the user should set the ‘preboot’ as follows:

  setenv preboot "if test -n $prebootcmd; then echo; echo Running pre-boot command; run prebootcmd;fi;**gpio set 7;gpio input 8;**"

8.3 U-Boot Upgrade

The U-Boot environment is created automatically by the U-Boot during the U-Boot upgrade process.

Upon U-Boot upgrade all the parameters in the U-Boot environment are overwritten by the upgraded U-Boot.

Therefore, it is recommended to save all modifications to the U-Boot environment made by the user prior to the U-Boot upgrade and apply them to the updated U-Boot’s environment.

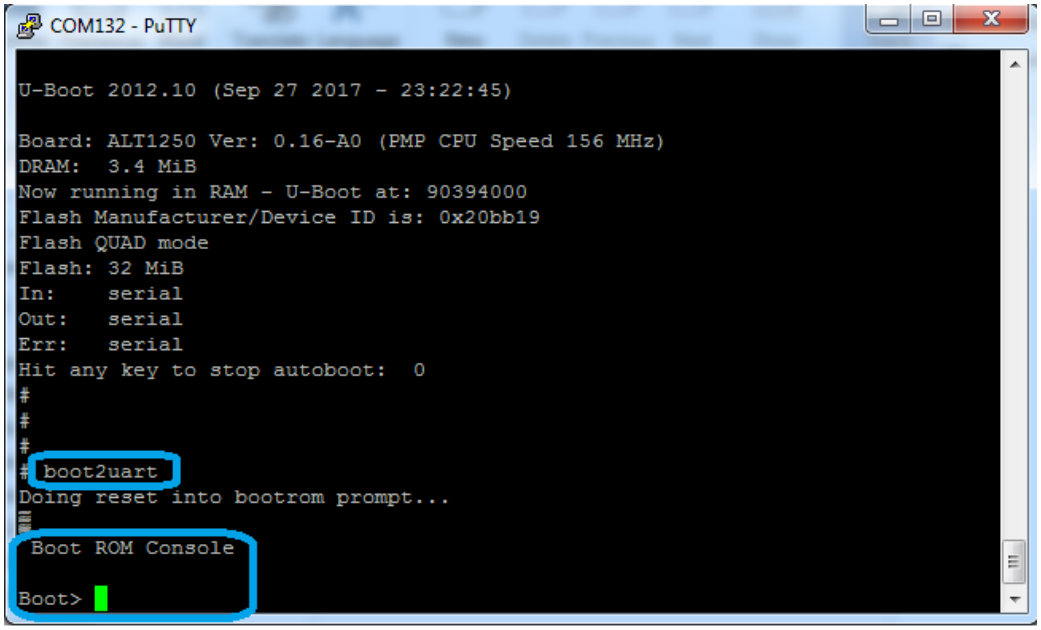
8.4 Booting in Boot-ROM State

The U-Boot supports the ability to boot in Boot-ROM state.

Booting in this state is required when the user wishes to burn a device via the ImageBurnTool in ‘Recovery’ mode.

In order to boot the device in Boot-ROM state, the user should run the ‘boot2uart’ command in a U-Boot terminal.

This command will trigger a device reset, after which the device will boot in Boot-ROM state (as can be seen in the screenshot below).



The ‘Boot>’ prompt indicates that the device is in Boot-ROM state.

9 Device Access over PPP Connection

Note: All UART ports references in this chapter are aligned to the UART ports assignment of Altair's ALT1250 Reference Board.

For customer designs which have different UART ports assignment than Altair's Reference Board, although the UART ports to be used are different, the operation and steps to be taken are the same as for Altair's Reference Board.

In case different / additional setting is required for customer designs, it will be stated.

The Reference Product supports the following access methods over a PPP connection:

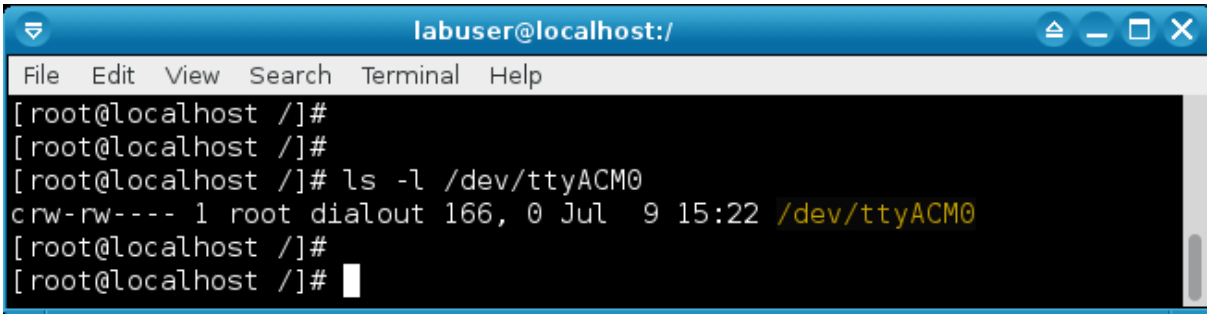
- Telnet – provides access to the CLI mechanism
- FTP – provides FTP read / write access to the OpenRTOS

9.1 Establishing PPP Connection

Currently, establishing the PPP connection is supported only on a Linux machine.

Below are the required steps for establishing the PPP connection:

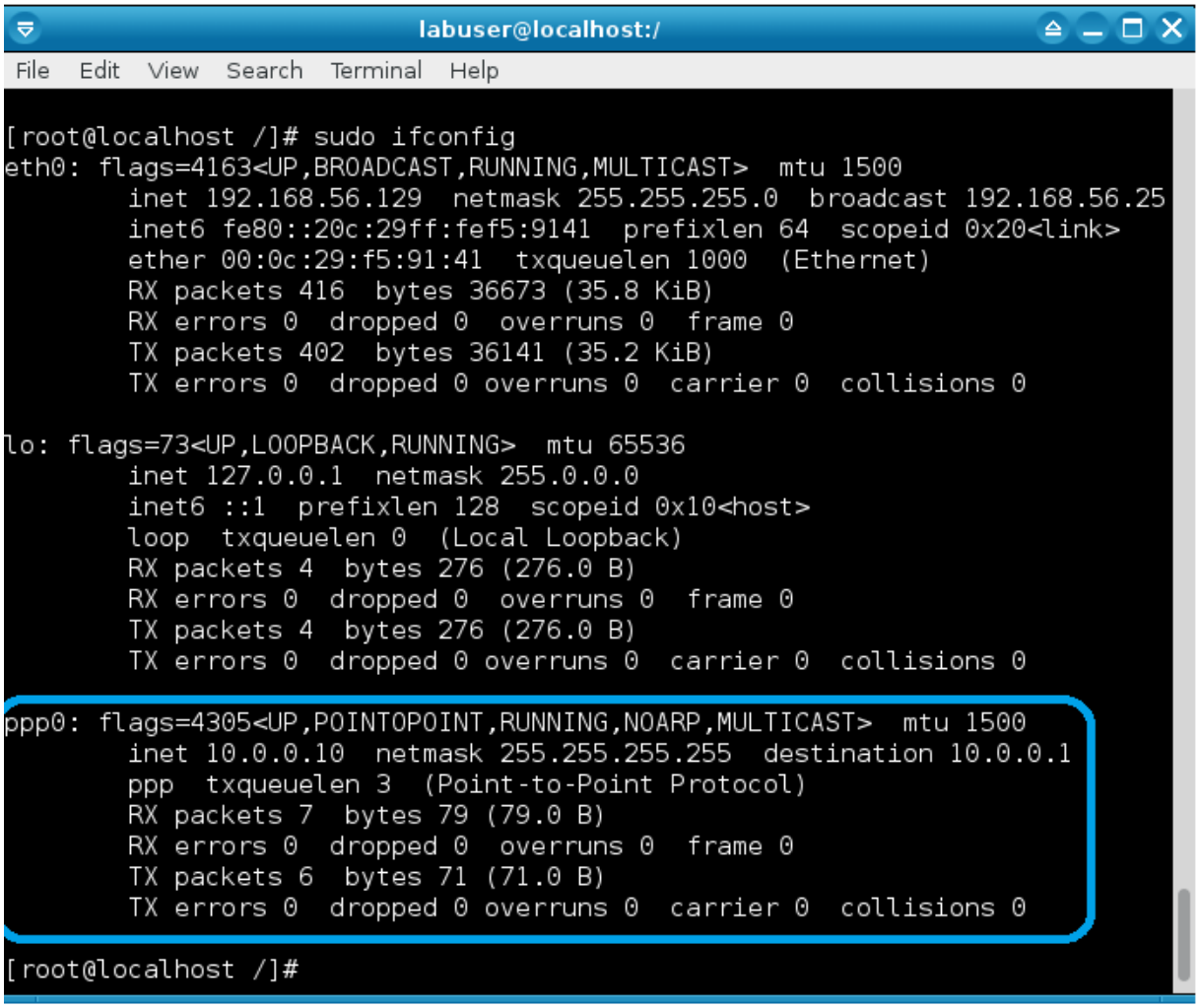
1. Connect a UART serial cable to the device's 'UART0/2' micro-USB connector.
2. In the Linux machine - open a terminal shell and verify that the 'ACM0' interface is identified by:
   ls -l /dev/ttyACM0



3. Open a new terminal shell on "/dev/ttyACM0" and run:
   ATDT
4. Close the terminal which was opened on "/dev/ttyACM0".
5. In the first terminal shell (opened in step #2), establish the PPP connection by:
   sudo pppd /dev/ttyACM0 115200 debug &
6. Verify the PPP interface ('ppp0') was created by:
   ifconfig

After establishing the PPP connection, the following IP addresses are assigned (as seen below):

- Linux host ('ppp0' inet) - 10.0.0.10
- Device ('ppp0' destination) - 10.0.0.1

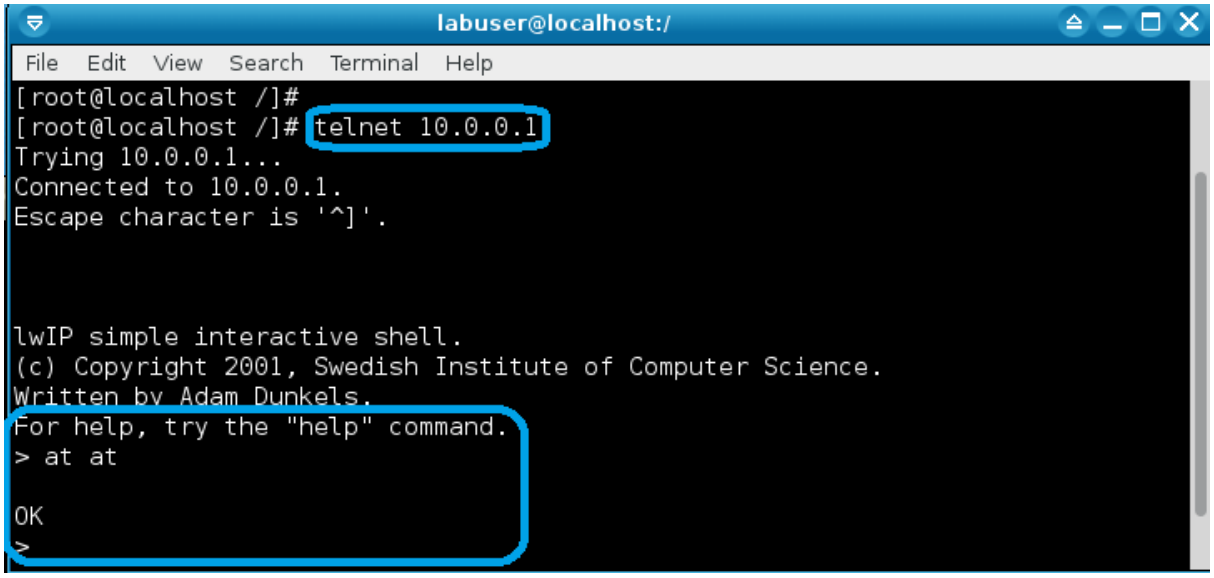


9.2 Connecting via Telnet

In the Linux machine's terminal shell - run:

telnet 10.0.0.1

Once the telnet connection is established, the user has CLI access to the device (as shown below).

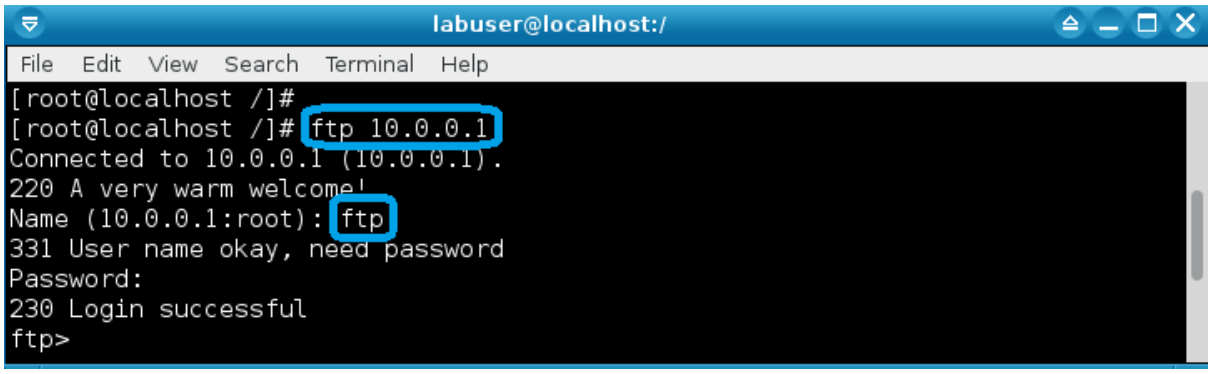


9.3 Connecting via FTP

In the Linux machine's terminal shell - run:

ftp 10.0.0.1

The username is 'ftp' and there's no password (just press 'Enter').

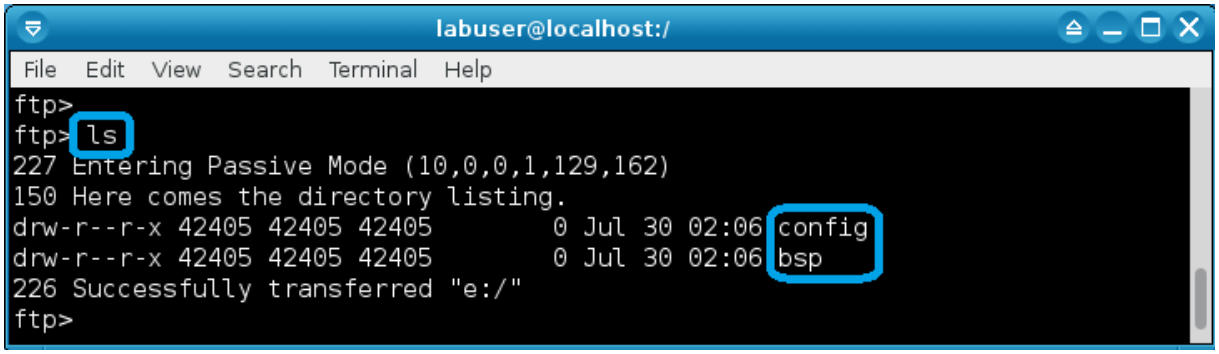


Once the FTP connection is established, the user has read / write access to the device's RAM which holds a mirror copy (created at boot time) of the file-system stored on the device's serial flash.

9.3.1 FTP Read Access

At boot time, the 'config' and 'bsp' folders are copied from the device's serial flash (from "**d**:/config" and "**d**:/bsp") to the device's RAM (to "**e**:/config" and "**e**:/bsp").

During run-time, the reading operations are made from the device's RAM (drive e:/).



9.3.2 FTP Write Access

During run-time, writing operations are made in the device's RAM (drive e:/) .

In order for the writing operations to be saved to the serial flash (drive d:/), the user needs to follow the steps below:

1. Since only the 'config' and 'bsp' folders are copied from the RAM to the serial flash, if the user writes to a file which is not in one of these folders, the file should be copied into either the 'config' or 'bsp' folders before saving it.
2. Reboot the device by the 'shutdown' CLI command or the 'atz' AT command.

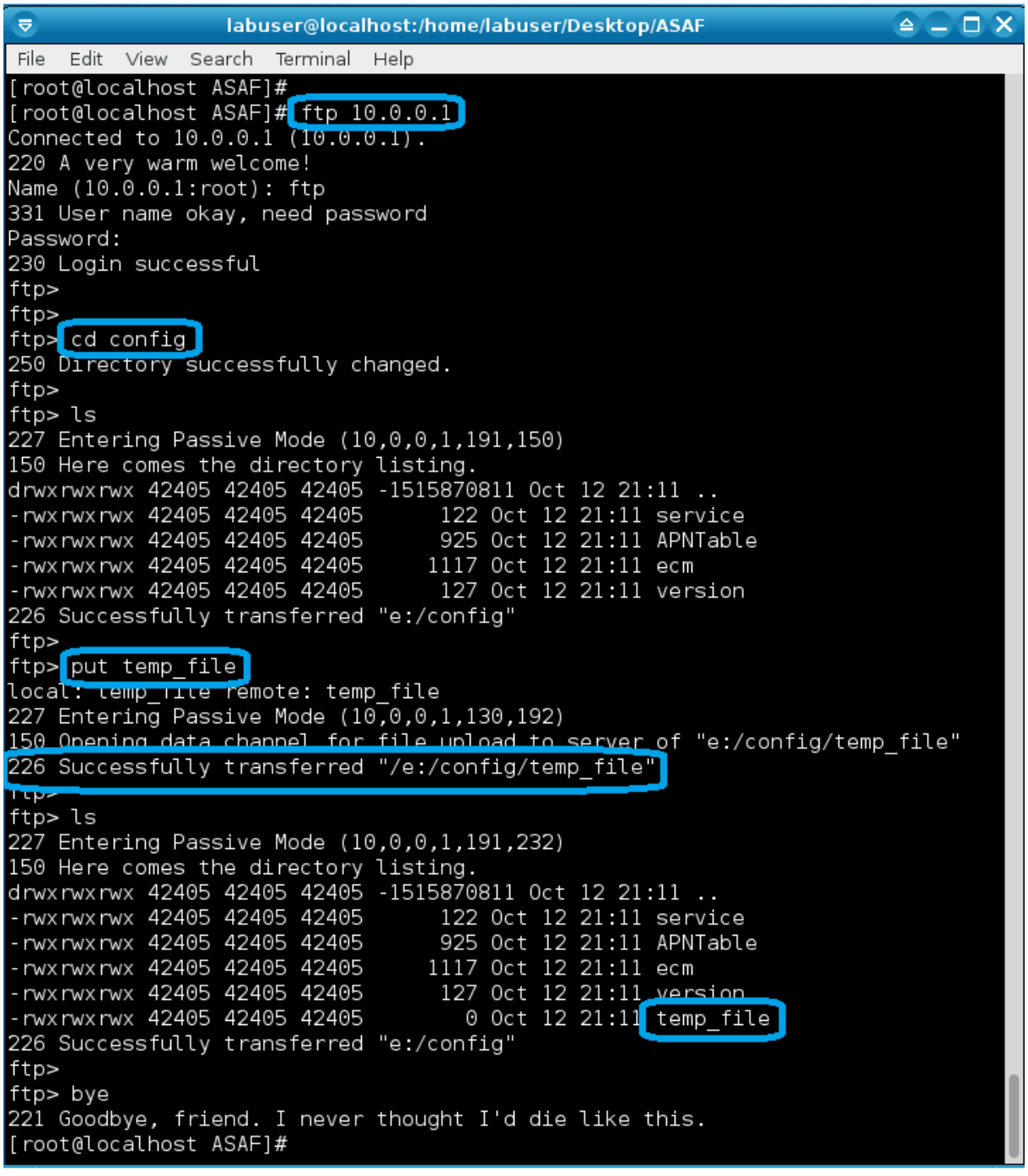
Upon a device reset by 'shutdown' or 'atz', the content of the 'config' and 'bsp' folders in the RAM will be copied into the 'config' and 'bsp' folders in the serial flash.

Example:

Below is an example of copying a new file ('tmp_file') from the Linux machine to the device's RAM (drive e:/), and writing it to the device's serial flash (drive d:/).

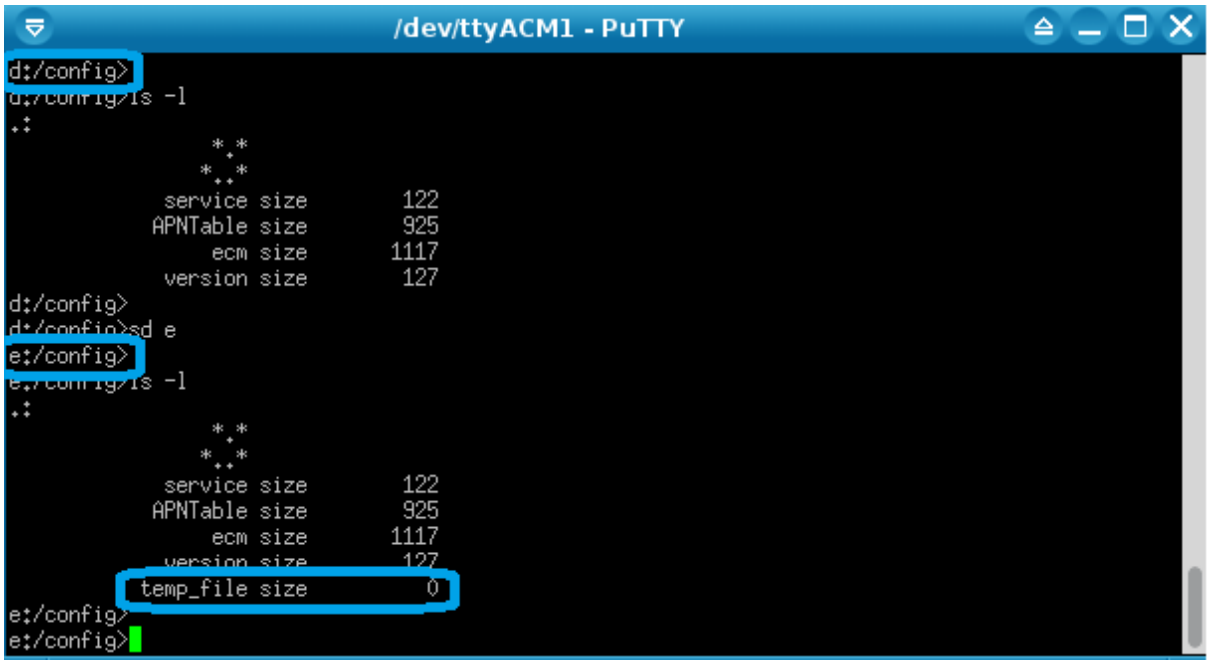
Step #1

Connecting to the device via FTP and copying the 'tmp_file' to the device's RAM under "e:/config/".



Step #2

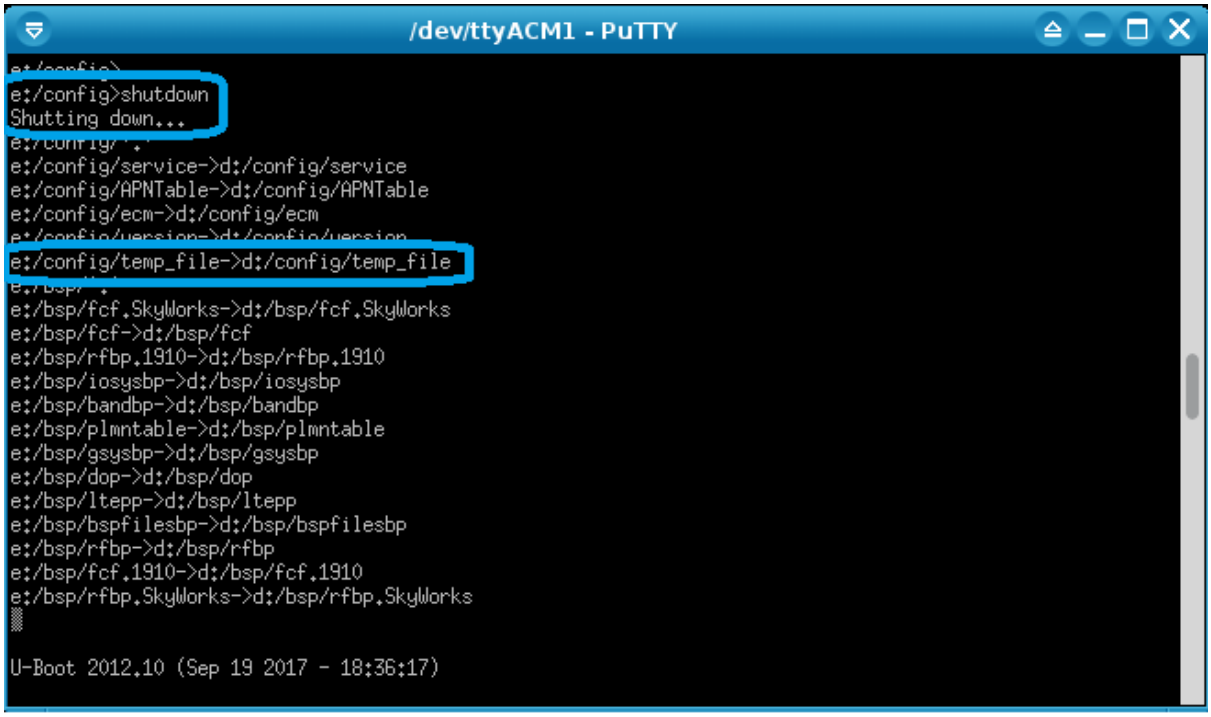
Via the device's CLI shell, it can be seen that 'tmp_file' does not exist in the serial flash under "d:/config/" but does exist in the device's RAM under "e:/config/".



Step #3

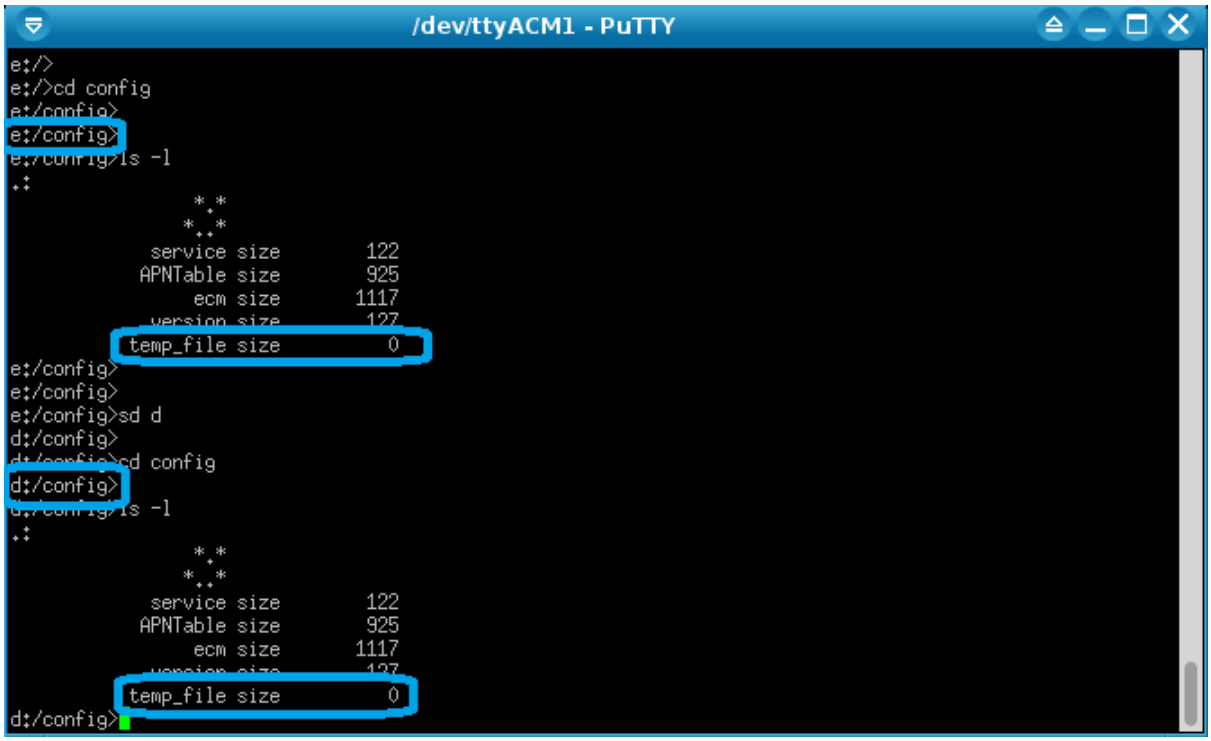
Running the 'shutdown' CLI command.

Before the device is reset, the content of "e:/config/" (including 'tmp_file') is copied from the RAM to the serial flash under "d:/config/".



Step #4

After the device boots, ‘tmp_file’ exists under “e:/config/” in the RAM and under “d:/config/” in the serial flash.



10 Switching UART Functionality at Run-time

Note: All UART ports references in this chapter are aligned to the UART ports assignment of Altair's ALT1250 Reference Board.

For customer designs which have different UART ports assignment than Altair's Reference Board, although the UART ports to be used are different, the operation and steps to be taken are the same as for Altair's Reference Board.

In case different / additional setting is required for customer designs, it will be stated.

The UART functionality on the AL1250 Reference Board is depicted in the table below:

Table 2. UART Ports Functionality

UART Port	Default Functionality	Optional Functionality
UART0	CLI mechanism	AT channel
UART1	Logging channel	
UART2	AT channel	PPP channel

10.1 UART0 Functionality

The user can switch between the functionalities of UART0 as follows (as can be seen in the screenshot below):

1. Connect a UART serial cable to the device's 'UART0/2' micro-USB connector.
2. Open a terminal shell to the CLI COM port (the default functionality of UART0 is CLI).
3. In order to switch to the AT channel functionality, the user needs to run the 'serialSwitch' command.
4. After step #3, the terminal will switch to an AT channel terminal and the user can run AT commands in it.
5. In order to switch back to CLI functionality, the user needs to run the 'ATDT' or 'ATO' AT command.
6. After step #5, the terminal will switch to a CLI terminal.

10.2 UART2 Functionality

The user can switch between the functionalities of UART2 as follows:

1. Connect a UART serial cable to the device's 'UART0/2' micro-USB connector.
2. Open a terminal shell to the AT channel COM port (the default functionality of UART2 is AT channel).
3. In order to switch to the PPP channel functionality, the user needs to run the 'ATDT' AT command.
4. After step #3, the AT channel will switch to a PPP channel, on which the user can establish a PPP connection.
5. In order to switch back to AT channel functionality, the user needs to run the '+++' command in the PPP terminal.
6. After step #5, the terminal will switch to an AT channel terminal.

11 BSP Files Access

Note: All UART ports references in this chapter are aligned to the UART ports assignment of Altair's ALT1250 Reference Board.

For customer designs which have different UART ports assignment than Altair's Reference Board, although the UART ports to be used are different, the operation and steps to be taken are the same as for Altair's Reference Board.

In case different / additional setting is required for customer designs, it will be stated.

The BSP files are located on the device's serial flash under "**d**:/bsp".

At boot time, the 'bsp' folder is copied from the device's serial flash to the device's RAM (to "**e**:/bsp").

Accessing the BSP files during run-time is done as follows:

- Reading BSP files – reading operations are made from the device's RAM (drive e:/).
- Writing BSP files – writing operations are divided to 2 steps:
  - Writing to the device's RAM (drive e:/) – the written BSP information is valid only for current run-time.
  - Writing to the device's serial flash (drive d:/) – the written BSP information is valid also after device reset.

Accessing the BSP files is done using the 'BspManager' tool as explained below.

Since the communication between the device and the 'BspManager' tool is conducted over the logging channel, the host PC must be connected to the device's 'UART1' micro-USB connector.

11.1 Pre-setting

11.1.1 Setting the Logging Port Parameters in the Device

Please follow the steps in the 'Setting the Logging Port Parameters in the Device' section under the 'FW Debug Logging' chapter.

11.1.2 Configuration Tool Setting

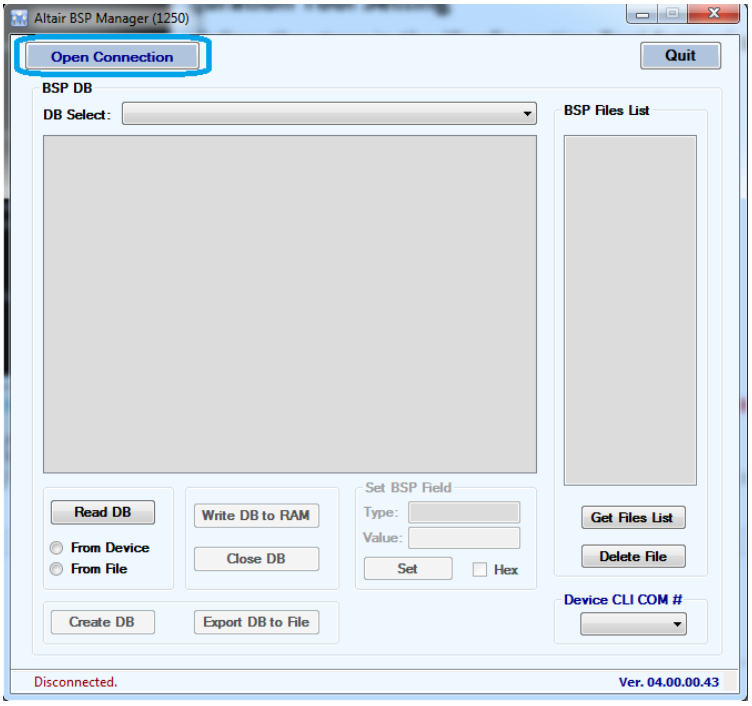
Please follow the steps in the 'Configuration Tool Setting' section under the 'FW Debug Logging' chapter.

11.2 Reading BSP Files

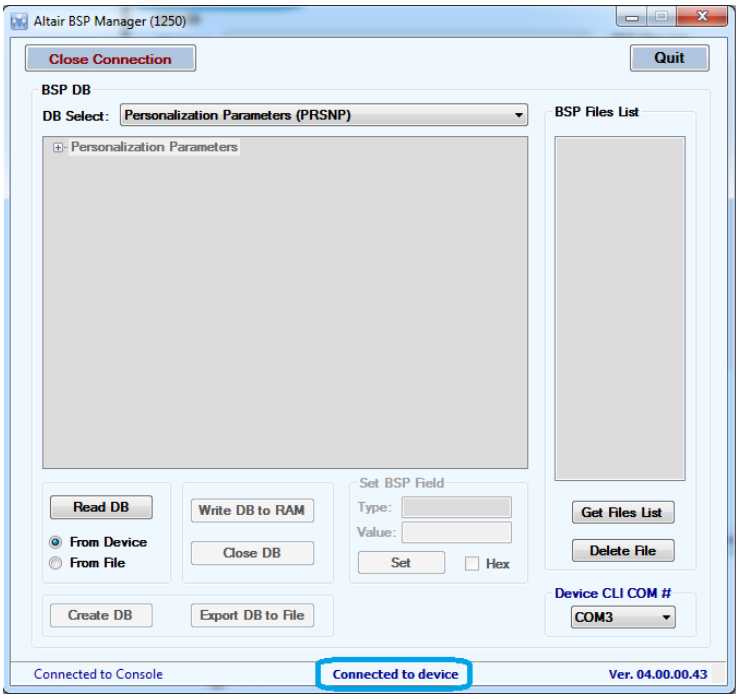
The 'BspManager' tool can be opened from:

"Start->All programs->Altair Semiconductor->PcTools"

1. Press the 'Open Connection' button as shown in the screenshot below.



2. Verify that “Connected to device” is shown in the tool’s status bar (bottom of the tool, as shown in the screenshot below).

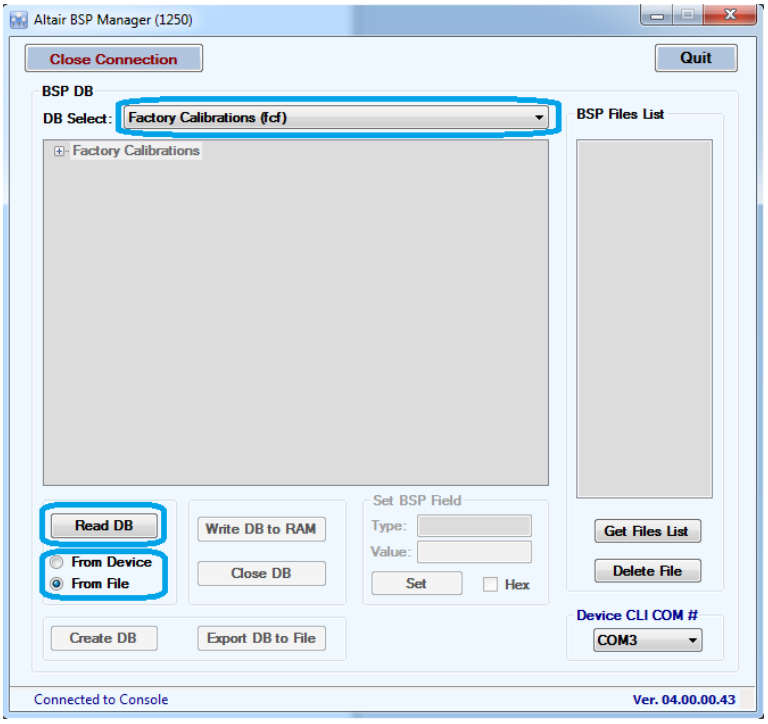


In case of failure to connect, please verify the device has booted properly and that the pre-setting requirements have been set properly.

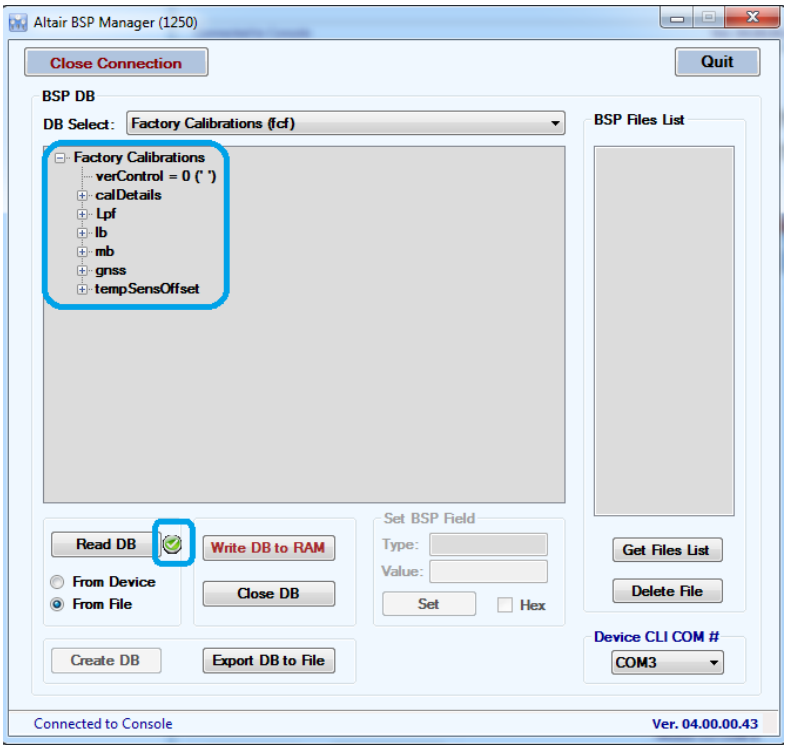
3. Choose the BSP file you wish to load in the ‘DB Select’ drop-down list.

After that, choose from where to load the BSP file (from the device or from a saved BSP file on the PC) by choosing the matching radio-button.

Finally, press the 'Read DB' button.



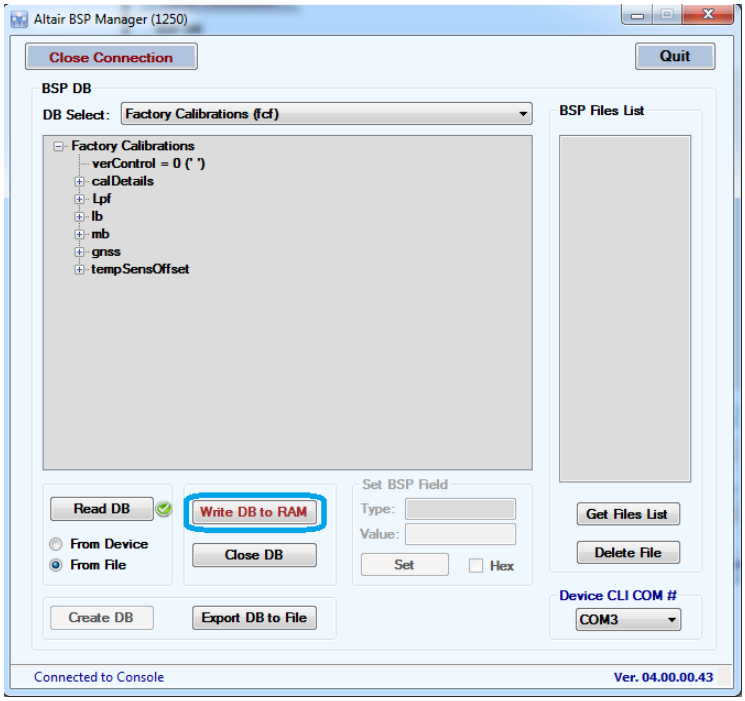
4. Verify that the BSP file has been opened properly, by the green 'V' indication next to the 'Read DB' button and the actual display of the BSP file content.



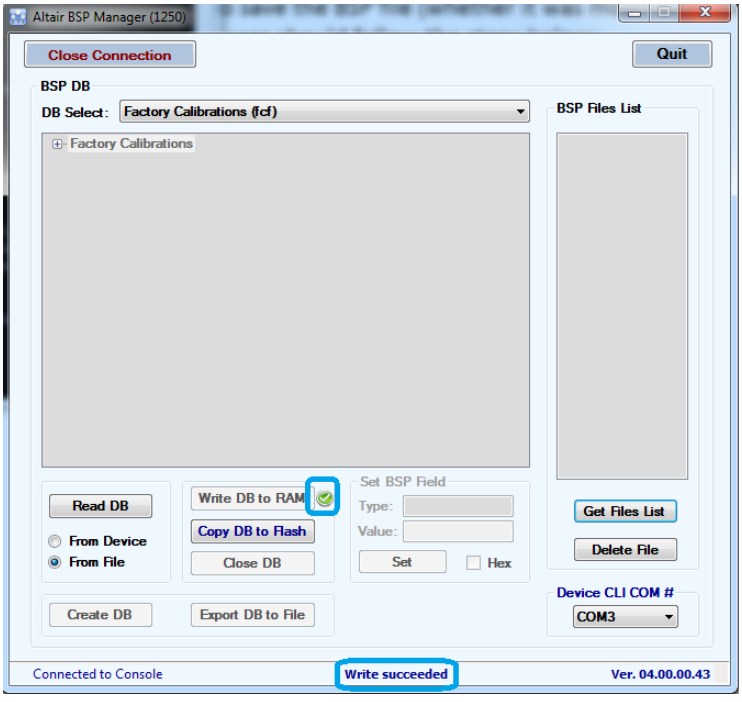
11.3 Writing BSP Files

In order to save the BSP file (whether it was modified or not) to the device's serial flash, the user should follow the steps below:

1. Write the BSP file to the device's RAM by pressing the 'Write DB to RAM' button.



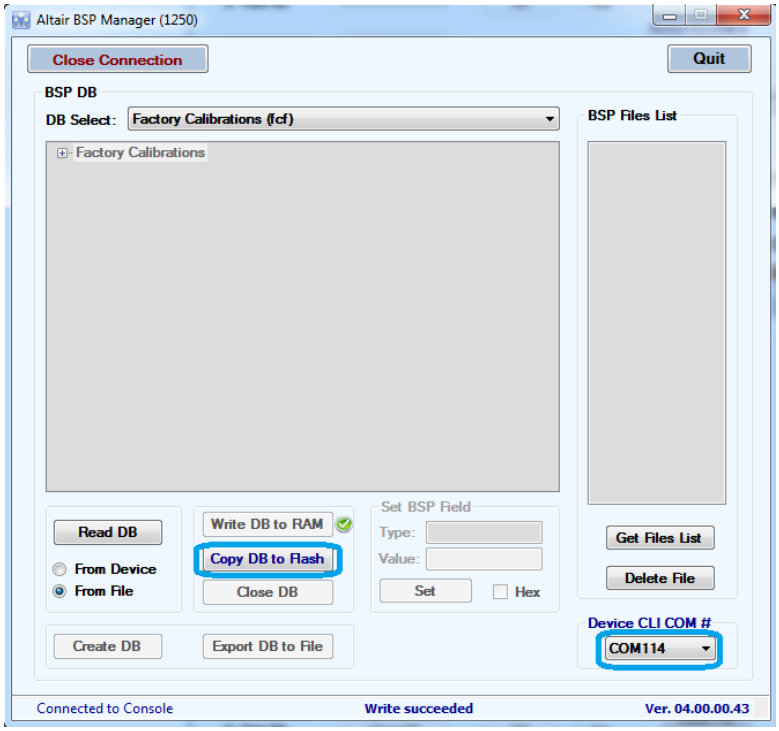
2. Verify that the writing succeeded, by the green 'V' indication next to the 'Write DB to RAM' button and the "Write succeeded" which is shown in the tool's status bar (bottom of the tool, as shown in the screenshot below).



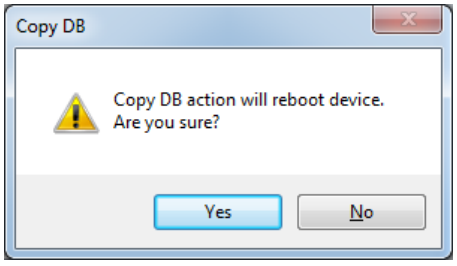
3. Write the BSP file from the device's RAM to the device's serial flash by pressing the 'Copy DB to Flash' button.

   Important note: Once the 'Copy DB to Flash' button is pressed, the BspManager sends the 'atz' AT command to the device (which will trigger a device reset). This command is sent over the CLI COM port, therefore the CLI COM port should be set in the 'Device CLI COM #' drop-down list prior to pressing the 'Copy DB to Flash' button.

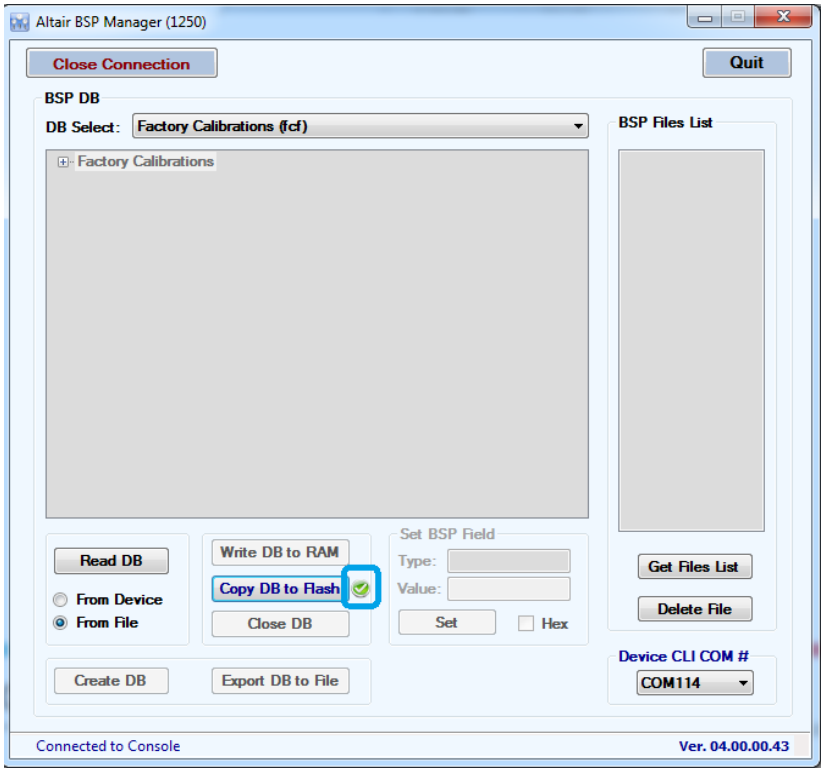
   If the CLI COM port is occupied by another application (e.g. a CLI terminal is open), the application should be closed prior to pressing the 'Copy DB to Flash' button.



4. Approve the device reset in the 'Copy DB' dialog-box.



5. Verify the writing to the device's serial flash was successful, by the green 'V' indication next to the 'Copy DB to Flash' button.


Altair BSP Manager (1250)
Close Connection
Quit
BSP DB
DB Select:
Factory Calibrations (fcf)
Factory Calibrations
BSP Files List
Read DB
From Device
From File
Write DB to RAM
Copy DB to Flash
Close DB
Set BSP Field
Type:
Value:
Set
Hex
Get Files List
Delete File
Create DB
Export DB to File
Device CLI COM #
COM114
Connected to Console
Ver. 04.00.00.43

12 Appendix A: ALT1250 Chipset Configuration

Different ALT1250 chipset revisions require different HW related settings in order to operate properly. Such configuration should be written in the device's OTP.

When working with ALT1250 **CS1.1**, the following should be set:

Note: This configuration is not required when working with ALT1250 **CS1.0.**

1. Open a terminal to the U-Boot at boot time.

   For accessing the U-Boot at boot time, please refer to the 'Accessing the U-Boot at Boot Time' section.

2. In the U-Boot terminal – run the following:

   otpsb 30308
   otpsb 30309
   otpsb 30311

3. Verify the configuration was written properly by running:

   otpdump 30304 8

   The expected output is:

   Step A

   3 2 1

   1098 7654 3210 9876 5432 1098 7654 3210

   .... 1011 0000 :30304 0x000000B0

4. Reset the device.

5. Open a terminal to the U-Boot at boot time.

   The following printout indicates that the device's HW revision was set to CS1.1:

   Board: ALT1250 Ver: 0.16-**B0**

   Before the setting, the printout was:

   Board: ALT1250 Ver: 0.16-**A0**