Juno ARM Development Platform

Version: 2.0

Getting Started Guide

ARM®

Juno ARM Development Platform

Getting Started Guide

Release Information

Document History

Issue	Date	Confidentiality	Change
A	02 July 2015	Non-Confidential	First release
B	29 July 2015	Non-Confidential	Second release
C	11 August 2015	Non-Confidential	Third release
D	17 September 2015	Non-Confidential	Fourth release
E	15 October 2015	Non-Confidential	Fifth release
F	22 December 2015	Non-Confidential	Sixth release

Federal Communications Commission Notice

This device is test equipment and consequently is exempt from part 15 of the FCC Rules under section 15.103 (c).

CE Declaration of Conformity

The system should be powered down when not in use.

It is recommended that ESD precautions be taken when handling Versatile™ Express boards.

The motherboard generates, uses, and can radiate radio frequency energy and may cause harmful interference to radio communications. There is no guarantee that interference will not occur in a particular installation. If this equipment causes harmful interference to radio or television reception, which can be determined by turning the equipment off or on, you are encouraged to try to correct the interference by one or more of the following measures:

- Ensure attached cables do not lie across the target board
- Reorient the receiving antenna
- Increase the distance between the equipment and the receiver
- Connect the equipment into an outlet on a circuit different from that to which the receiver is connected
- Consult the dealer or an experienced radio/TV technician for help

Note

It is recommended that wherever possible shielded interface cables be used.

Contents

Juno ARM Development Platform Getting Started Guide

Preface

About this book 7

Feedback 9

Chapter 1 Introduction

1.1 About the Juno ADP 1-11

1.2 Ports and I/O 1-12

1.3 Use of Ethernet ports 1-13

1.4 UART configuration 1-14

1.5 Monitor compatibility 1-15

Chapter 2 Configuration

2.1 Connecting to a host computer 2-17

2.2 SD card filesystem layout 2-18

2.3 Flash memory programming 2-19

2.4 Erasing the flash memory 2-21

2.5 Board recovery procedure 2-22

2.6 Juno SoC configuration 2-23

2.7 Linux filesystems 2-24

Chapter 3 Firmware

3.1 Firmware components 3-26

3.2 System Control Processor (SCP) firmware 3-27
3.3 Application Processor (AP) firmware 3-28

Appendix A Revisions
A.1 Revisions Appx-A-30

Preface

This preface introduces the *Juno ARM Development Platform Getting Started Guide*.

It contains the following:

- *About this book* on page 7.
- *Feedback* on page 9.

About this book

This book describes the Juno *ARM® Development Platform* (ADP). The Juno ADP is an ARM Versatile™ Express V2M-Juno motherboard containing a Juno ADP SoC and reference software.

Using this book

This book is organized into the following chapters:

Chapter 1 Introduction
This chapter introduces the Juno *ARM Development Platform* (ADP).

Chapter 2 Configuration
This chapter describes the configuration of the V2M-Juno motherboard.

Chapter 3 Firmware
This chapter describes the firmware that the Juno ADP supports.

Appendix A Revisions
This appendix describes the technical changes between released issues of this book.

Glossary

The ARM Glossary is a list of terms used in ARM documentation, together with definitions for those terms. The ARM Glossary does not contain terms that are industry standard unless the ARM meaning differs from the generally accepted meaning.

See the *ARM Glossary* for more information.

Typographic conventions

italic
Introduces special terminology, denotes cross-references, and citations.

bold
Highlights interface elements, such as menu names. Denotes signal names. Also used for terms in descriptive lists, where appropriate.

`monospace`
Denotes text that you can enter at the keyboard, such as commands, file and program names, and source code.

`monospace`
Denotes a permitted abbreviation for a command or option. You can enter the underlined text instead of the full command or option name.

`monospace italic`
Denotes arguments to monospace text where the argument is to be replaced by a specific value.

`monospace bold`
Denotes language keywords when used outside example code.

`<and>`
Encloses replaceable terms for assembler syntax where they appear in code or code fragments. For example:

```
MRC p15, 0, <Rd>, <CRn>, <CRm>, <Opcode_2>
```

SMALL CAPITALS
Used in body text for a few terms that have specific technical meanings, that are defined in the *ARM glossary*. For example, IMPLEMENTATION DEFINED, IMPLEMENTATION SPECIFIC, UNKNOWN, and UNPREDICTABLE.

Additional reading

See *http://infocenter.arm.com*, for access to ARM documentation.

ARM® publications

This book contains information that is specific to this product. See the following documents for other relevant information:

- *Juno ARM® Development Platform (ADP) SoC Technical Reference Manual* (ARM DDI 0515).
- *Juno ARM® Development Platform SoC Technical Overview* (ARM DTO 0038).
- *ARM® Compute Subsystem SCP Message Handling Interface* (ARM DUI 0922).
- *Power State Coordination Interface (PSCI) System Software on ARM® Systems* (ARM DEN 0022).
- *Trusted Board Boot Requirements CLIENT (TTBR-CLIENT) System Software on ARM* (ARM DEN 0006C-1).
- *ARM® Compute Subsystem SCP Message Interface Protocols* (ARM DUI 0922).
- *ARM® Versatile™ Express Juno r2 Development Platform (V2M-Juno r2) Technical Reference Manual* (100114).
- *ARM® Versatile™ Express Juno r1 Development Platform (V2M-Juno r1) Technical Reference Manual* (100122).
- *ARM® Versatile™ Express Juno Development Platform (V2M-Juno) Technical Reference Manual* (100113).
- *ARM Technical Support Knowledge Articles*.
- *Support and Maintenance*.

Feedback

Feedback on this product

If you have any comments or suggestions about this product, contact your supplier and give:

- The product name.
- The product revision or version.
- An explanation with as much information as you can provide. Include symptoms and diagnostic procedures if appropriate.

Feedback on content

If you have comments on content then send an e-mail to *errata@arm.com*. Give:

- The title *Juno ARM Development Platform Getting Started Guide*.
- The number ARM DEN0928F.
- If applicable, the page number(s) to which your comments refer.
- A concise explanation of your comments.

ARM also welcomes general suggestions for additions and improvements.

———— **Note** ————

ARM tests the PDF only in Adobe Acrobat and Acrobat Reader, and cannot guarantee the quality of the represented document when used with any other PDF reader.

————————————

Chapter 1
Introduction

This chapter introduces the Juno *ARM Development Platform* (ADP).

It contains the following sections:

- *1.1 About the Juno ADP* on page 1-11.
- *1.2 Ports and I/O* on page 1-12.
- *1.3 Use of Ethernet ports* on page 1-13.
- *1.4 UART configuration* on page 1-14.
- *1.5 Monitor compatibility* on page 1-15.

1.1 About the Juno ADP

The Juno ADP is a 64-bit software development platform consisting of an ARM Versatile Express V2M-Juno motherboard containing a Juno ADP SoC and reference software.

The ADP is based around an ARMv8-A *Compute Subsystem* (CSS) and provides:

- **Juno r0 and r1**
  Dual core Cortex®-A57 MPCore processors (big cluster).
  Juno r2
  Dual core Cortex-A72 MPCore processors (big cluster).
- Quad core Cortex-A53 MPCore processors (LITTLE cluster).
- A Mali™-T624 series GPU with four shader cores for 3D Graphics acceleration and GPU compute.
- A Cortex M3-based *System Control Processor* (SCP) which offloads power control and thermal management.
- 8GB of DDR3-1600 memory (dual channel, striped).
- An architecture that is aligned with Level 1 of the Server Base System Architecture.
- ARM big.LITTLE MP support. On Juno r1 only, the LITTLE cluster is limited to the lowest DVFS *Operating Performance Point* (OPP) due to hardware limitations.

  Note

  The big.LITTLE MP scheduler is not optimized.

The following hardware features are available only on Juno r1 and later:

- *Peripheral Component Interconnect Express* (PCIe) Gen 2.0 support.

Reference firmware and other software images are available for the Juno ADP from Linaro.

Note

The use of Juno software is subject to the terms of the *Juno End-User License Agreement*.

External interfaces include:

- Support for LogicTile Express boards that you can use to extend the Juno ADP.
- USB 2.0 and a custom SoC to *Field Programmable Gate Array* (FPGA) prototyping extension interface.

The V2M-Juno motherboard is available from ARM. For information, see *www.arm.com/juno*.

Note

The V2M-Juno motherboard is lead-free. The ARM Connected Community pages also provide additional information for users of the Juno ADP, including FAQs and instructions on using the Linaro software deliverables. For more information, see *https://community.arm.com/groups/arm-development-platforms*.

1.2 Ports and I/O

The following figure shows the front panel of the V2M-Juno motherboard.

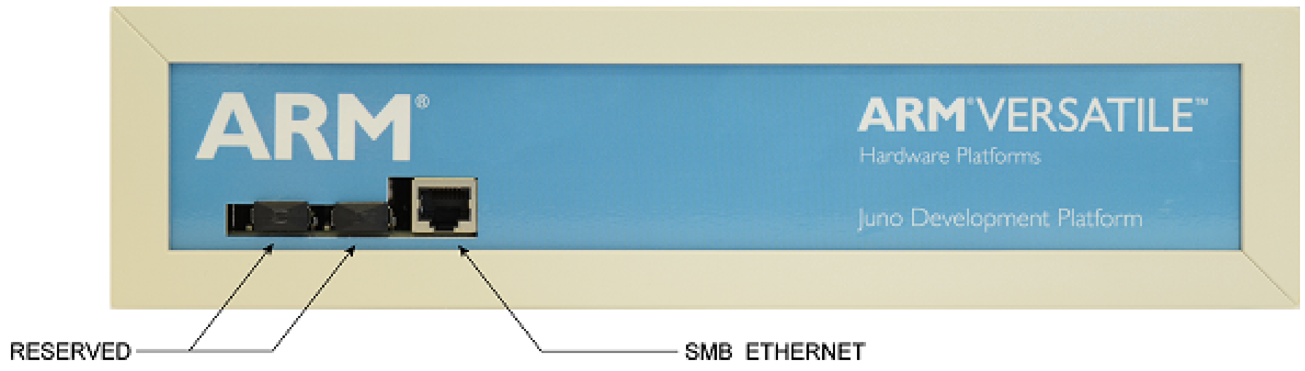

Figure 1-1 V2M-Juno motherboard front panel

The front Ethernet port is enabled in hardware on all current versions of Juno. It is connected internally over a *Static Memory Bus* (SMB). The USB ports exist only on Juno r0.

The following figure shows the rear panel of the V2M-Juno motherboard.

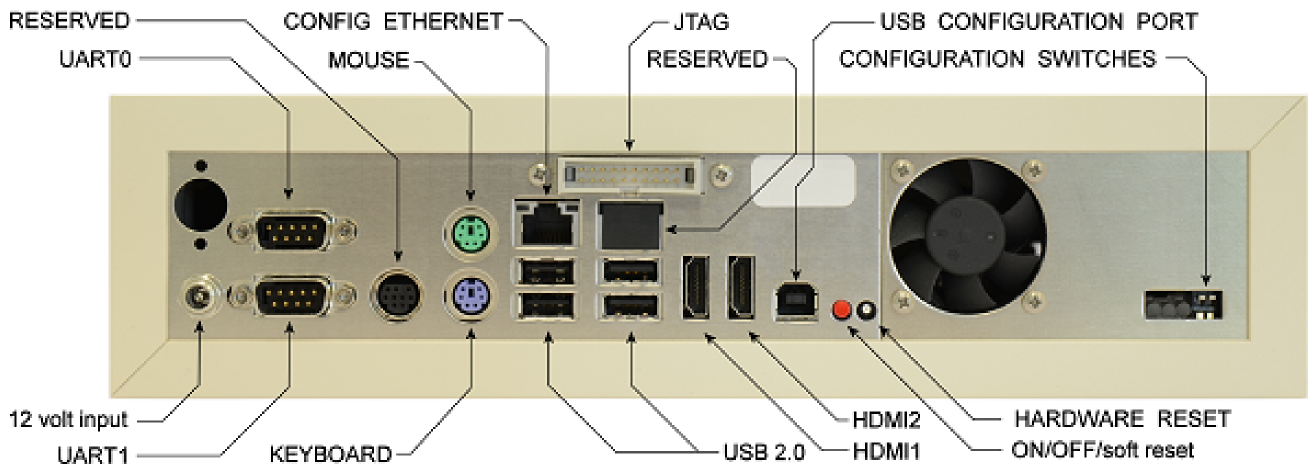

Figure 1-2 V2M-Juno motherboard rear panel

The rear Ethernet port is not enabled in hardware on Juno r0. On Juno r1 and later, it is enabled and connected over the PCI Express bus.

1.3 Use of Ethernet ports

This section describes the Ethernet ports on the rear panel of the V2M-Juno motherboard.

1.3.1 Boot support

If using *Trivial File Transfer Protocol* (TFTP) boot with *Unified Extensible Firmware Interface* (UEFI), the front Ethernet port must be used.

U-Boot does not contain support for booting over an Ethernet connection.

1.3.2 Port numbering in Linux and Android with UEFI

UEFI is responsible for passing the MAC address of the primary Ethernet port to the kernel using its command line. UEFI reads this address from the *System Configuration and Control* (SCC) registers.

Juno r0

The front Ethernet port is numbered eth0. The rear port is not enabled in hardware and therefore the kernel does not allocate a number.

Juno r1 and later

Using the Linaro tracking kernel with the OpenEmbedded filesystem results in the rear Ethernet port being numbered eth0. This kernel and filesystem combination does not support the front Ethernet port without extra configuration.
The *Linaro Stable Kernel* (LSK) does not support the rear Ethernet port. The front Ethernet port is therefore numbered eth0.

1.3.3 Port numbering in Linux and Android with U-boot

If U-Boot is used, you must use the `ifconfig` tool within Linux to set up an Ethernet port as follows:

```
ifconfig eth0 hw ether [MAC address]
```

1.4 UART configuration

There are 4 UARTs on the V2M-Juno motherboard. Two (SoC UART0 and SoC UART1) are on the rear panel, while two are only exposed as headers.

SoC UART0
On the back panel.
Used by the boot firmware, for example the V2M-Juno motherboard, ARM Trusted Firmware, and UEFI.
It also used by the Operating System, for example, the Linux kernel.

SoC UART1
On the back panel.
Used by the runtime firmware, for example, ARM Trusted Firmware or UEFI.

FGPA UART0
Corresponds to the J55 header on the V2M-Juno motherboard.
Contact ARM for further information about this type of header.
Used by the ARM Trusted Firmware.

FGPA UART1
Corresponds to the J56 header on the V2M-Juno motherboard.
Contact ARM for further information about this type of header.
Used by the SCP Firmware if it is built in debug mode.

1.4.1 Settings

For all UARTs, the settings are:

Baud rate	115200
Data bits	8
Parity bits	None
Stop bits	1

Related concepts

1.2 Ports and I/O on page 1-12.

1.5 Monitor compatibility

There are hardware limitations affecting HDMI support and issues might be encountered when using the V2M-Juno motherboard with certain monitors.

- On some monitors, the native resolution is not always achieved, and the V2M-Juno motherboard output degrades to 1024×768. On the next reboot, there is a chance that the native resolution might be selected.
- For some display modes on some monitors, the display can periodically lose synchronization, causing a brief picture loss each time this occurs. The ratio of display time to blanking time varies significantly across monitors and modes.
- On some monitors, the V2M-Juno motherboard is unable to establish a working display mode.

The following table summarizes the results for monitors that ARM has tested.

Table 1-1 Monitor compatibility with the V2M-Juno motherboard

Manufacturer	Model	Native resolution	Native resolution achieved		Fallback to 1024x768 @ 60Hz achieved	No display achieved	
			Success rate	**Blanking occurs**	**Success rate**		**Blanking occurs**
BenQ	GL2460	1920×1080@60Hz	91%	Yes[a]	0%	n/a	9%
BenQ	GL2450	1920×1080@60Hz	100%	Yes[a]	n/a	n/a	n/a
BenQ	G2200WT	1680×1050@60Hz	100%	No	n/a	n/a	n/a
BenQ	BL2201	1680×1050@60Hz	100%	No	n/a	n/a	n/a
Dell	P1911	1440×900@60Hz	100%	No	n/a	n/a	n/a
HP	L1940T	1280×1024@60Hz	100%	No	n/a	n/a	n/a
Iiyama	ProLite E511S	1600×1200@60Hz	100%	No	n/a	n/a	n/a
LG	LED 22M35	1920×1080@60Hz	100%	Yes[a]	n/a	n/a	n/a
NEC	EA274WMi	2560×1440@52.1Hz	100%	Yes[a]	n/a	n/a	n/a
NEC	LCD2070NX	1600×1200@60Hz	20%	No	n/a	n/a	n/a
Samsung	SyncMaster S22B370	1920×1080@60Hz	0%	n/a	0%	n/a	
Samsung	SyncMaster SA850	2560×1440@60Hz	0%	n/a	100%[b]	No	n/a

[a] No blanking occurs when invoking the kernel with a video command-line option that includes an R flag. For example:

```
video=HDMI-A-1:1920x1080R@60
```

[b] The Samsung SyncMaster SA850 is available for purchase at the time of writing and works well when using 1024×768 or half of the native resolution, but only if the following kernel command-line option is provided to force the resolution. ARM recommends adding this option to get the monitor to work:

```
video=HDMI-A-1:1280x720@60
```

(If you have a 3.10 based kernel, then replace `HDMI-A-1` with `DVI-D-1`)

Chapter 2
Configuration

This chapter describes the configuration of the V2M-Juno motherboard.

It contains the following sections:

- *2.1 Connecting to a host computer* on page 2-17.
- *2.2 SD card filesystem layout* on page 2-18.
- *2.3 Flash memory programming* on page 2-19.
- *2.4 Erasing the flash memory* on page 2-21.
- *2.5 Board recovery procedure* on page 2-22.
- *2.6 Juno SoC configuration* on page 2-23.
- *2.7 Linux filesystems* on page 2-24.

2.1 Connecting to a host computer

A set of files that are stored on an SD card on the board determines the configuration of the V2M-Juno motherboard. The SD card can be accessed using a USB-B socket on the rear panel of the board. When connected to a host computer, the SD card appears as a USB mass storage device with a FAT16 filesystem. The files in the filesystem are edited to control the configuration of the V2M-Juno motherboard.

Follow these steps:

1. Connect a serial terminal to the top 9-pin UART0 connector on the rear panel.
2. Connect a USB cable between the USB-B connector on the rear panel and a USB port of your host computer.
3. Connect the 12 volt power supply to the V2M-Juno motherboard.
   The serial terminal shows the command prompt `Cmd>`.

   ——— **Note** ———

   If the command prompt is not shown, ensure that both configuration switches are in the top position. Also reset the V2M-Juno motherboard using the hardware reset button. The hardware reset button is on the rear panel.

   ————————

4. At the `Cmd>` prompt on the serial terminal, issue the command `usb_on`.

   ```
   Cmd> usb_on
   ```

   The configuration SD card is now visible on the host computer as a mass storage device.

2.2 SD card filesystem layout

If you are using the recovery images or following the Linaro software release instructions, you do not need to change any of the firmware or configuration files.

The V2M-Juno motherboard firmware images, SoC software images, and settings are stored on an SD card on the board. At boot time, the V2M-Juno motherboard firmware accesses the SD card and copies certain images into internal flash memory.

The SD card is exposed to a host computer as a USB mass storage device with a FAT16 filesystem. A brief description of the filesystem layout follows.

```
./
│
├── SITE1         - Motherboard firmware (BIOS, IOFPGA image, PMIC configuration)
│     ├──HBI0262B  - Configuration files for Juno r0 only
│     ├──HBI0262C  - Configuration files for Juno r1 only
│     └──HBI0262D  - Configuration files for Juno r2 only
├── SITE2         - Supporting files for LogicTile daughterboards
├── SOFTWARE      - SoC software images (SCP Firmware, ARM Trusted Firmware,
│                   UEFI, Linux, etc)
└── config.txt    - Configuration file for the V2M-motherboard
```

Figure 2-1 filesystem layout

Related concepts

1.2 Ports and I/O on page 1-12.

2.3 Flash memory programming

This section describes flash memory programming.

——— **Note** ———

If you are using the recovery images or following the Linaro software release instructions, the firmware or configuration files do not need to be changed.

———

For an image file to be programmed into the V2M-Juno motherboard flash memory, it must first be placed onto the SD card filesystem. Each image must also have a corresponding entry in the `SITE1/HBI0262[B/C/D]/images.txt` file. This file is used by the V2M-Juno motherboard firmware to determine which images to program into flash memory at boot time.

——— **Note** ———

ARM recommends that all SoC software images to be loaded are placed in the `SOFTWARE` directory of the SD card.

———

The following example illustrates the required format for the `images.txt file`. The number of entries is likely to differ on a newly received V2M-Juno motherboard.

Example 2-1 Images.txt file format

```
TITLE: Versatile Express Images Configuration File

[IMAGES]
TOTALIMAGES: 2                     ;Number of Images (Max: 32)

NOR0UPDATE: AUTO                   ;Image Update:NONE/AUTO/FORCE
NOR0ADDRESS: 0x00000000            ;Image Flash Address
NOR0FILE: \SOFTWARE\fip.bin        ;Image File Name
NOR0LOAD: 00000000                 ;Image Load Address
NOR0ENTRY: 00000000                ;Image Entry Point

NOR1UPDATE: AUTO                   ;Image Update:NONE/AUTO/FORCE
NOR1ADDRESS: 0x03EC0000            ;Image Flash Address
NOR1FILE: \SOFTWARE\bl1.bin        ;Image File Name
NOR1LOAD: 00000000                 ;Image Load Address
NOR1ENTRY: 00000000                ;Image Entry Point
```

The `TOTALIMAGES` value informs the V2M-Juno motherboard firmware of the number of images to be programmed into the flash memory of the board. It must match the number of entries that follow it. If the value is too low, that is, smaller than the number of images, some images are not programmed into the flash memory. If it is too high, then the behavior of the V2M-Juno motherboard firmware is UNDEFINED.

——— **Note** ———

The numbering for images begins at zero. For example, `NOR0UPDATE`, `NOR1UPDATE`.

———

Each entry must have a consistent format that matches the examples that are given. The `NOR[x]` naming prefix must increment with every new entry.

You can define a maximum of 32 entries, with `NOR31` being the highest possible prefix name. Above this value, the behavior of the motherboard firmware is UNDEFINED.

For each `NOR[x]` image, the following fields are required:

UPDATE

None

The file is programmed into flash memory if it is not already present. On subsequent boots the file in flash is not updated, even if it has changed on the SD card.

Auto

The V2M-Juno motherboard firmware determines if the file on the SD card is consistent with the file in flash memory. If not, the flash memory is updated.

Force

The V2M-Juno motherboard firmware programs the flash memory with the contents of the file on the SD card at every boot, regardless of whether it has changed or not.

ADDRESS

The base address within the flash memory that the image is programmed to. The address must always be prefixed with `0x`.

NAME

The path to, and name of, the file that is programmed. All paths must be specified from the root of the SD card filesystem.

LOAD

Leave as zero unless required.

ENTRY

Leave as zero unless required.

2.4 Erasing the flash memory

If you have problems with your Juno software and have to completely reset the images in flash you must erase everything that is there first. It is good practice to do this when applying a recovery image or moving to a new Linaro release in case any configuration requires changing. The following steps assume that the V2M-Juno motherboard is connected to a host computer.

1. At the `Cmd>` prompt on the serial terminal, issue the command `flash`.
2. The V2M-Juno motherboard will power on automatically and there is a brief delay while the V2M-Juno motherboard firmware initializes.
3. At the `Flash>` prompt, enter `eraseall`.
4. Once the V2M-Juno motherboard firmware has finished erasing the flash memory, at the `Flash>` prompt, enter `exit`.
5. The V2M-Juno motherboard powers down and returns to the `Cmd>` prompt.

On the subsequent boot, the V2M-Juno motherboard firmware copies images from the SD card into the flash memory, following the definitions in the `images.txt` file. You should see the following:

```
Cmd> flash

Switching on main power...
PMIC RAM configuration (pms_v104.bin)...
IOFPGA  config: PASSED

Flash> eraseall
Erasing flash device
Erasing Flash
........................
Erasing Flash
....

Flash> exit
Switching off ATX PSU.

Cmd>
```

Related concepts

2.1 Connecting to a host computer on page 2-17.

2.5 Board recovery procedure

This section describes how to reset the settings and firmware images of the V2M-Juno motherboard to a default state using a bundle of pre-built images that are provided by Linaro.

1. Ensure that the V2M-Juno motherboard is first connected to a host computer.
2. (Optional) Save any existing files on the SD card that you want to retain to the host computer.
3. Erase the flash memory of the V2M-Juno motherboard. Do not boot the board until the subsequent steps have been completed, otherwise the flash memory is programmed again prematurely.
4. Format the SD card.

   The procedure for formatting the SD card depends on the operating system of the host computer and is outside the scope of this document. Alternatively, recursively remove all files and directories on the SD card filesystem, beginning at the top-level directory.
5. Extract the prebuilt Linaro binaries following the instructions at *https://community.arm.com/groups/arm-development-platforms*.

   There are multiple firmware bundles available, each of which provides a different software environment, for example, Android, OpenEmbedded, or Busybox.

   Extract the required bundle directly onto the SD card, ensuring that the directory structure of its archive is preserved.
6. If the host computer operating system is Linux, ARM recommends that you use the sync command to ensure that any pending write operations have been flushed.
7. Safely eject the mass storage device when the archive extraction has completed.
8. Power on the board by issuing the `reboot` command at the `Cmd>` prompt on the serial terminal. Alternatively, press the red ON/OFF/Soft reset button on the rear panel.
9. The V2M-Juno motherboard firmware copies the appropriate images into the flash memory. If the version of the V2M-Juno motherboard BIOS image has changed, then the BIOS update progress is also shown on this initial boot (but not after).

Related concepts

1.2 Ports and I/O on page 1-12.

2.1 Connecting to a host computer on page 2-17.

2.4 Erasing the flash memory on page 2-21.

2.6 Juno SoC configuration

Some of the configuration settings for the Juno SoC are exposed through the `board.txt` file located in `SITE1/HBI0262[B/C/D]`. These include settings for PLL frequencies, clocks, and entry point addresses for the SCP and AP ROM firmware images.

The following example shows a typical `board.txt` file.

Note

ARM recommends that you do not modify any values in this file unless you have a specific need to do so. Incorrect values can result in instability or an inability to boot the SCP Firmware and ARM Trusted Firmware successfully.

Example 2-2 board.txt file

```
BOARD: HBI0262
TITLE: V2M-Juno DevChip Configuration File
[SCC REGISTERS]
TOTALSCCS: 9
SCC: 0x10C 0x00019300 ;A53 PLL Register 1
SCC: 0x0F8 0x0BEC0000 ;BL1 entry point
SCC: 0xA14 0x00000000 ;PCLKDBG_CONTROL DIV=1
SCC: 0x118 0x003F1000 ;SYS PLL Register 0 (1600MHz)
SCC: 0x11C 0x0001F100 ;SYS PLL Register 1
SCC: 0x100 0x003F1000 ;A57/A72 PLL Register 0 (800MHz)
SCC: 0x104 0x0001F300 ;A57/A72 PLL Register 1
SCC: 0x0FC 0xABE40000 ;BL0 entry point
SCC: 0x108 0x00331000 ;A53 PLL Register 0 (650MHz)
```

2.7 Linux filesystems

By default, the V2M-Juno motherboard does not contain any flash Linux filesystems, for example Android, or Open Embedded, anywhere in its on-board storage. To fully boot either of these platforms it is necessary to attach a filesystem, for example, Android or OpenEmbedded, using an external storage device. If no filesystem is attached at boot time, and the Linux kernel that is programmed into flash expects one, the kernel will boot as far as possible before stopping while it waits for a filesystem to be attached.

Note

If you are using a pre-built Busybox image, the system will boot Linux fully without the need for an external USB storage device containing a filesystem.

ARM recommends that a USB disk is connected using one of the four USB 2.0 ports on the rear panel. The type of USB disk can be a flash drive or a hard drive as both are compatible.

See *http://community.arm.com/groups/arm-development-platforms* for details on preparing an Android or OpenEmbedded filesystem image and how to copy it onto a USB disk ready for use.

Related concepts

1.2 Ports and I/O on page 1-12.

Chapter 3
Firmware

This chapter describes the firmware that the Juno ADP supports.

It contains the following sections:

- *3.1 Firmware components* on page 3-26.
- *3.2 System Control Processor (SCP) firmware* on page 3-27.
- *3.3 Application Processor (AP) firmware* on page 3-28.

3.1 Firmware components

The Juno ADP is supported by several pieces of firmware. These are preloaded on the V2M-Juno motherboard or can be obtained as pre-built image bundles from the Linaro software releases.

They include:

V2M-Juno motherboard microcontroller (MCC) firmware
The V2M-Juno MCC is a microcontroller on the motherboard that takes care of early board setup before the SCP or the application processors within the Juno SoC are powered on. The MCC is also responsible for managing firmware updates by copying images into the on-board flash memory.

System Control Processor (SCP) firmware
The SCP is a dedicated Cortex-M3 within the Juno SoC that provides low-level power management and system control for the Juno platform. The SCP provides a messaging interface to the application processors so that application firmware and software can make requests of it. This interface is described in the *ARM® Compute Subsystem SCP Message Interface Protocols* document, available at *http://infocenter.arm.com/help/topic/com.arm.doc.dui0922-/index.html*.

ARM Trusted Firmware
ARM Trusted Firmware provides a reference implementation of Secure world software for ARMv8-A, including Exception level 3 (EL3) software.

Unified Extensible Firmware Interface (UEFI)
The Juno ADP implementation of UEFI provides operating system loader support and runtime services. It is based on the EFI Development Kit 2 (EDK2) implementation available from the Tianocore project at *http://www.tianocore.org*.

U-Boot firmware
An alternative loader provided with certain pre-built Linaro image bundles that is used instead of UEFI. The implementation is based on U-Boot from DENX.

——— Note ———

ARM strongly recommends that you immediately upgrade to the latest available firmware before using the V2M-Juno motherboard.

———

Related concepts

2.5 Board recovery procedure on page 2-22.

3.3.1 ARM Trusted Firmware on page 3-28.

3.3.2 Unified extensible Interface Firmware (UEFI) on page 3-28.

3.2 System Control Processor (SCP) firmware

The System Control Processor (SCP) manages the overall power, clock, reset, and system control of the ADP. Because of the hardware design, the SCP firmware is an inherently trusted part of the ADP software system. All the memory that the SCP uses for execution and private storage is on-chip to prevent attackers tampering with it.

Note

The SCP Firmware is only available as a pre-built binary.

3.2.1 SCP ROM firmware

The SCP ROM firmware is the first code to execute on the Juno ADP after a cold reset. This code is fixed for the lifetime of the device and therefore executes minimal code to maximize robustness and reduce the risk of security vulnerabilities.

The SCP ROM firmware configures the initial state of the hardware platform, for example:

- Cores that are released from reset.
- Clocks that are running and their default frequencies.
- Power domains within the SoC which are powered on.

Boot protocol

The SCP ROM firmware interacts with the ARM Trusted Firmware, which securely transfers the SCP RAM firmware image to the SCP at runtime. The SCP then passes control from the SCP ROM firmware to the SCP RAM firmware and the boot process continues.

More information about the Boot Over MHU (BOM) protocol used for this process is available as part of the *ARM Compute Subsystems SCP Message Interface Protocols* document, available from *http://infocenter.arm.com/help/topic/com.arm.doc.dui0922-/index.html*.

3.2.2 SCP RAM firmware

The SCP RAM firmware is a second firmware image for the SCP which provides runtime services to the application processors.

3.2.3 System Control and Power Interface (SCPI)

The SCPI is the generic runtime interface to the SCP from the AP through the MHU.

It includes:

- Reporting the capabilities of the system and certain devices within it. For example, reporting the available sensors, or the number of power domains.
- Control of power domains and voltages.
- Control of PLLs and clock frequencies.
- Control of the performance level of the processors and GPU through *Dynamic Voltage and Frequency Scaling* (DVFS).
- Watchdog and timer services.
- Sensor monitoring, thermal cut-out, and fault reporting.

This interface is described in described in the *ARM Compute Subsystems SCP Message Interface Protocols* document, available from *http://infocenter.arm.com/help/topic/com.arm.doc.dui0922-/index.html*.

3.3 Application Processor (AP) firmware

The AP firmware comprises ARM Trusted Firmware and a choice of either UEFI or U-Boot.

Note

The ARM deliverables only contain partial support for security and virtualization, and third-party software vendors must work with the supplied software to integrate commercial trusted operating systems and hypervisors, if required.

3.3.1 ARM Trusted Firmware

ARM Trusted Firmware provides a reference implementation of Secure World software for ARMv8-A, including Exception Level 3 (EL3) software. It provides implementations of various ARM interface standards, such as the *Power State Coordination Interface* (PSCI), *Trusted Board Boot Requirements* (TBBR) and Secure monitor code.

For more information on ARM Trusted Firmware, see *https://github.com/ARM-software/arm-trusted-firmware*. You can obtain a validated source SHA by following the release code syncing instructions at *http://community.arm.com/groups/arm-development-platforms*.

The ARM Trusted Firmware code is designed for reuse or porting to other ARMv8-A model and hardware platforms.

3.3.2 Unified extensible Interface Firmware (UEFI)

The *Unified Extensible Firmware Interface* (UEFI) is a boot firmware specification that the UEFI forum maintains and develops. ARM is a member of the UEFI forum and contributes to the UEFI ARM bindings.

Linaro provides an AArch64 implementation of this specification for ADPs, based on the EFI Development Kit 2 (EDK2), available from Tianocore at *http://www.tianocore.org*.

UEFI supports:
- Booting an Operating System from NOR Flash or USB mass storage.
- Ethernet and PXE boot.
- ACPI 5.1.

3.3.3 U-boot firmware

An alternative loader provided with certain pre-built Linaro image bundles that is used instead of UEFI.

The implementation is based on U-Boot from DENX, available from *http://www.denx.de/wiki/U-Boot*.

Appendix A
Revisions

This appendix describes the technical changes between released issues of this book.

It contains the following sections:

- *A.1 Revisions* on page Appx-A-30.

A.1 Revisions

This appendix describes the technical changes between released issues of this book.

Table A-1 Table A-1 Issue A

Change	Location	Affects
First release.	-	-

Table A-2 Table A-2 Differences between Issue A and Issue B

Change	Location	Affects
Removed Product Release Status section.	*Chapter 1 Introduction* on page 1-10	Issue B
IPA support statement removed.	*Chapter 1 Introduction* on page 1-10	Issue B
Use of Ethernet ports on page 1-4 added.	*1.3 Use of Ethernet ports* on page 1-13	Issue B
Updated UART descriptions.	*1.4 UART configuration* on page 1-14	Issue B

Table A-3 Table A-3 Differences between Issue B and Issue C

Change	Location	Affects
Updated links to Linaro site.	Throughout	Issue C
Link to Connected Community added.	*1.1 About the Juno ADP* on page 1-11	Issue C
Firmware bundles archive text removed.	*2.5 Board recovery procedure* on page 2-22	Issue C

Table A-4 Table A-4 Differences between Issue C and Issue D

Change	Location	Affects
Reorganized footnotes by deleting the original footnote “a”.	*1.5 Monitor compatibility* on page 1-15	Issue D
Updated 3.18 kernel command-line argument, replacing DVI-D with HDMI-A.	*1.5 Monitor compatibility* on page 1-15	Issue D
Deleted footnote reference for BenQ GL2450 in the “Success rate” column.	*1.5 Monitor compatibility* on page 1-15	Issue D

Table A-5 Table A-5 Differences between Issue D and Issue E

Change	Location	Affects
No changes made.	-	Issue E

Table A-6 Table A-6 Differences between Issue E and Issue F

Change	Location	Affects
Converted to DITA.	Throughout	Issue F
Clarified content within section 1.1.	*1.1 About the Juno ADP* on page 1-11	Issue F
Updated text following figure 1-1.	*1.2 Ports and I/O* on page 1-12	Issue F
Removed redundant text from short description.	*1.3 Use of Ethernet ports* on page 1-13	Issue F
Clarified success rate.	*1.5 Monitor compatibility* on page 1-15	Issue F
Updated figure 2-1.	*2.2 SD card filesystem layout* on page 2-18	Issue F
Clarified file path.	*2.3 Flash memory programming* on page 2-19	Issue F
Updated file path and example 2-2.	*2.6 Juno SoC configuration* on page 2-23	Issue F