

Armadillo-400 Series Software Manual

**Version 1.7.0
2012/02/29**

Atmark Techno, Inc.

Armadillo Site

Armadillo-400 Series Software Manual

Atmark Techno, Inc.

060-0035 AFT Bldg., N5E2, Chuo-ku, Sapporo
TEL 011-207-6550 FAX 011-207-6570

Copyright © 2010-2012 Atmark Techno, Inc.

Version 1.7.0
2012/02/29

Table of Contents

1. Preface	10
1.1. Document and Related Files Versions	11
1.2. Who Should Read This Document	11
1.3. Document Structure	11
1.4. Typographical Conventions	11
1.4.1. Fonts	11
1.4.2. Command Entry Examples	11
1.4.3. Icons	12
1.5. Acknowledgements	12
2. Precautions	13
2.1. Safety Precautions	13
2.2. Handling Precautions	14
2.3. Software Usage Precautions	14
2.4. Write Prohibited Regions	15
2.5. Electromagnetic Interference	15
2.6. Warranty	15
2.7. Exporting	16
2.8. Trademarks	16
3. System Overview	17
3.1. Armadillo-400 Series Basic Specifications	17
3.2. Basic Specifications of Armadillo-420 Basic Model	20
3.3. Basic Specifications of Armadillo-420 WLAN Model (AWL12 Compatible)	22
3.4. Basic Specifications of Armadillo-420 WLAN Model (AWL13 Compatible)	24
3.5. Basic Specifications of Armadillo-440 LCD Model	27
3.6. Basic Specifications of Armadillo-460 Basic Model	31
3.7. Memory Map	34
3.8. Software Make-up	35
3.8.1. Bootloader	35
3.8.2. Kernel	35
3.8.3. Userland	35
3.8.4. Downloader	36
3.9. Boot Modes	36
4. Before Getting Started	38
4.1. Preparation	38
4.2. Connections	38
4.3. Serial Console Software Configuration	43
5. Development Environment Set-up	44
5.1. Installing Cross Development Environment Packages	44
5.2. Installing Packages Required for Atmark-Dist Builds	45
5.3. Installing Cross Development Library Packages	45
6. Rewriting Flash Memory	47
6.1. Flash Memory Writing Regions	47
6.2. Installing Downloader	48
6.2.1. For Linux	48
6.2.2. For Windows	48
6.3. Rewriting Flash Memory with a Downloader	48
6.3.1. Preparation	49
6.3.2. For Linux	49
6.3.3. For Windows	49
6.4. Rewriting Flash Memory with tftpd1	50
6.5. Rewriting Flash Memory with netflash	51
6.6. Restoring Bootloader to Factory State	52
6.6.1. Preparation	52

6.6.2. For Linux	52
6.6.3. For Windows	53
6.7. Restoring Bootloader Parameters to Factory State	56
7. Building	58
7.1. Building Kernel and Userland Images	58
7.1.1. Preparing Source Code	58
7.1.2. Applying Default Configuration	60
7.1.3. Building	61
7.1.4. Customizing Images	62
7.1.5. Adding an Application to the Userland Image	66
7.2. Building Bootloader Images	67
7.2.1. Preparing Source Code	67
7.2.2. Building	67
8. Kernel and Userland Placement	68
8.1. Loading from a TFTP Server	68
8.1.1. File Placement	68
8.1.2. Boot Options	68
8.2. Loading from Storage	69
8.2.1. Partitioning	69
8.2.2. Creating Filesystems	70
8.2.3. Kernel Image Placement	71
8.2.4. Creating a Root Filesystem	71
8.2.5. Boot Device and Kernel Parameter Settings	73
9. Linux Kernel Device Driver Specifications	75
9.1. UART	75
9.2. Ethernet	77
9.3. MMC/SD/SDIO Host	77
9.4. USB 2.0 Host	78
9.5. Frame Buffer	78
9.6. LED Backlight	79
9.7. Touchscreen	79
9.8. Audio	80
9.9. GPIO	81
9.9.1. GPIO sysfs	81
9.9.2. Armadillo-200 Series Compatible GPIO Driver	84
9.10. LED	85
9.10.1. LED Class	85
9.10.2. Armadillo-200 Series Compatible LED Driver	86
9.11. Buttons	87
9.12. Real-time Clock	88
9.12.1. Enabled Real-Time Clock Selection	89
9.12.2. Alarm Interrupts	89
9.13. Watchdog Timer	91
9.14. I2C	91
9.15. SPI	92
9.16. One Wire	92
9.17. PWM	93
9.18. CAN	93
9.19. Keypad	95
9.20. Power Management	96
9.20.1. Sleep Function	96
9.20.2. PMIC Power Off Function	99
10. Armadillo-460 Expansion Bus	100
10.1. Kernel Configuration Options	100
10.1.1. Expansion Bus Operating Modes	100

10.1.2. Direct CPU Bus Mode (Synchronous) Only	100
10.1.3. Direct CPU Bus Mode (Asynchronous) Only	101
10.2. Memory Map	101
10.3. Precautions When Using ISA Drivers	103
10.4. Header File for Armadillo-460 PC/104	104
10.4.1. I/O Port Access Macros	104
10.4.2. Interrupt Number Conversion Marcos	104
A. Hermit-At Bootloader	105
A.1. version	105
A.1.1. version Example	106
A.2. info	106
A.2.1. info Example	106
A.3. memmap	106
A.3.1. memmap Example	107
A.4. mac	107
A.4.1. mac Example	107
A.5. md5sum	107
A.5.1. md5sum Example	107
A.6. erase	108
A.6.1. erase Example	108
A.7. setenv and clearenv	108
A.7.1. setenv/clearenv Example	109
A.7.2. Linux Kernel Parameters	109
A.8. setbootdevice	109
A.8.1. setbootdevice Example	109
A.9. frob	110
A.10. tftpd	110
A.10.1. tftpd Example	111
A.11. tftpboot	111
A.11.1. tftpboot Example	112
A.12. boot	112
A.12.1. boot Example	113
A.13. Note on Versions	113

List of Figures

3.1. Armadillo-420/440 Block Diagram	18
3.2. Armadillo-460 Block Diagram	19
3.3. Basic Layout of Armadillo-420 Basic Model	21
3.4. Basic Layout of Armadillo-420 WLAN Model (AWL12 Compatible)	23
3.5. Basic Layout of Armadillo-420 WLAN Model (AWL13 Compatible)	25
3.6. Basic Layout of Armadillo-440 LCD Model	28
3.7. Basic Layout of Armadillo-460 Basic Model	32
4.1. Armadillo-420 Basic Model Connections	39
4.2. Armadillo-420 WLAN Model (AWL12 Compatible) Connections	40
4.3. Armadillo-420 WLAN Model (AWL13 Compatible) Connections	41
4.4. Armadillo-440 LCD Model Connections	42
4.5. Armadillo-460 Basic Model Connections	43
5.1. Install Command	44
5.2. Installed Version Display Command	45
5.3. Cross Development Library Package Creation	46
5.4. Installing Cross Development Library Packages	46
5.5. apt-cross Command	46
6.1. Installing Downloader (Linux)	48
6.2. Download Command	49
6.3. Download Command (with Port Option)	49
6.4. Download Command (Unprotected)	49
6.5. Hermit-At Win32: Download Window	50
6.6. Hermit-At Win32: Download Dialog	50
6.7. tftpd Command Example	51
6.8. netflash Command Example	52
6.9. shoehorn Command Example	52
6.10. shoehorn Command Log	53
6.11. Bootloader Write Command Example	53
6.12. Hermit-At Win32: Shoehorn Window	54
6.13. Hermit-At Win32: shoehorn Dialog	54
6.14. Hermit-At Win32: Erase Window	55
6.15. Hermit-At Win32 : Erase Dialog	55
6.16. Hermit-At Win32: Download Window (After Erase)	56
6.17. Hermit-At Win32: Download Dialog (bootloader)	56
6.18. Returning Kernel Parameters to Default State	57
6.19. Returning Boot Device to Default Configuration	57
7.1. Source Code Preparation (Atmark-Dist)	58
7.2. Source Code Preparation (Linux Kernel)	59
7.3. Source Code Preparation (Aerial Driver)	59
7.4. Source Code Preparation (AWL13 Driver)	60
7.5. Atmark-Dist Build	62
7.6. Atmark-Dist Configuration	62
7.7. menuconfig: Main Menu	62
7.8. menuconfig: Kernel/Library/Defaults Selection	63
7.9. menuconfig: Do you wish to save your new kernel configuration?	64
7.10. menuconfig: Linux Kernel Configuration	65
7.11. Enabling AWLAN	65
7.12. menuconfig: Userland Configuration	66
7.13. Userland Image Customization	67
7.14. Hermit-At Source Archive Extraction	67
7.15. Hermit-At Build Example	67
8.1. tftpboot Command	68
8.2. tftpboot Command Example	69

8.3. Partitioning Procedure	70
8.4. Filesystem Creation Procedure	71
8.5. Kernel Image Placement	71
8.6. Root Filesystem Creation with Debian Archives	72
8.7. Root Filesystem Creation with Atmark-Dist Image	73
8.8. fstab Alter Example	73
8.9. Boot Device Designation	73
8.10. Root Filesystem Designation Example	74
9.1. GPIO sysfs Interrupt Sample Program	83
9.2. Alarm Interrupt Time Setting Example	90
9.3. I2C Transmission Speed Configuration	91
9.4. CAN Transmission Speed Calculation	95
10.1. Including the PC/104 Header File	104
A.1. version Syntax	106
A.2. version Example	106
A.3. info Syntax	106
A.4. info Example	106
A.5. memmap Syntax	106
A.6. memmap Example	107
A.7. mac Syntax	107
A.8. mac Example	107
A.9. md5sum Syntax	107
A.10. md5sum Example	108
A.11. erase Syntax	108
A.12. erase Example	108
A.13. setenv/clearenv Syntax	108
A.14. setenv and clearenv Example	109
A.15. setbootdevice Syntax	109
A.16. Assigning Flash Memory As Boot Device	110
A.17. Assigning TFTP Server As Boot Device	110
A.18. Assign SD/MMC Card As Boot Device	110
A.19. tftpd Syntax	110
A.20. tftpd Example	111
A.21. tftpboot Syntax	111
A.22. tftpboot Example	112
A.23. boot Syntax	112
A.24. boot Example	113

List of Tables

- 1.1. Armadillo-400 Series Models 10
- 1.2. Models and Corresponding Manuals 10
- 1.3. Fonts 11
- 1.4. Relationship Between Prompt and Execution Environment 11
- 1.5. Abbreviations Used In Command Entry Examples 12
- 3.1. Armadillo-400 Series Basic Specifications 17
- 3.2. Basic Specifications of RTC Option Module 20
- 3.3. Default States of Armadillo-420 Basic Model Expansion Interfaces 21
- 3.4. Basic Specifications of WLAN Option Module (AWL12 Compatible) 22
- 3.5. Default States of Armadillo-420 WLAN Model (AWL12 Compatible) Expansion Interfaces 23
- 3.6. Basic Specifications of WLAN Option Module (AWL13 Compatible) 24
- 3.7. Default States of Armadillo-420 WLAN Model (AWL13 Compatible) Expansion Interfaces 25
- 3.8. Basic Specifications of Expansion Board 27
- 3.9. Default States of Armadillo-440 LCD Model Expansion Interfaces 29
- 3.10. Default States of Armadillo-460 Basic Model Expansion Interfaces 32
- 3.11. Armadillo-420 Flash Memory Map 34
- 3.12. Armadillo-440/460 Flash Memory Memory Map 35
- 3.13. Jumper Settings 36
- 4.1. Serial Communication Configuration 43
- 5.1. List of Packages Required for Atmark-Dist Builds 45
- 6.1. Region Names and Corresponding Image Files 47
- 6.2. Downloader List 48
- 6.3. Region Names and Corresponding Options 51
- 6.4. Region Names and Corresponding Device Files 52
- 6.5. Bootloader Parameters 56
- 7.1. Product Name List 60
- 8.1. Kernel Image Download URLs 71
- 8.2. Debian Archive Download URLs 72
- 8.3. Atmark-Dist Image Download URL 72
- 9.1. Default Available UART 75
- 9.2. Serial Interface Device Files 76
- 9.3. UART Configuration 76
- 9.4. UART Configuration (Armadillo-460 Specific) 76
- 9.5. Ethernet Configuration 77
- 9.6. MMC/SD/SDIO Host Controller Configuration 77
- 9.7. USB Host Configuration 78
- 9.8. Frame Buffer Device Files 79
- 9.9. Frame Buffer Configuration 79
- 9.10. LED Backlight Configuration 79
- 9.11. Touchscreen Events 79
- 9.12. Touchscreen Configuration 80
- 9.13. Audio Configuration 81
- 9.14. GPIO_NAME and GPIO Pins 81
- 9.15. GPIO I/O Direction Configuration 82
- 9.16. GPIO Interrupt Type Configuration 82
- 9.17. GPIO sysfs Configuration 84
- 9.18. Armadillo-200 Series Compatible GPIO Driver GPIO List 84
- 9.19. Armadillo-200 Series Compatible GPIO Driver Device File 85
- 9.20. Armadillo-200 Series Compatible GPIO Driver ioctl Commands 85
- 9.21. Armadillo-200 Series Compatible GPIO Driver Configuration 85
- 9.22. LED List 86
- 9.23. LED Class Configuration 86
- 9.24. LED Node 86

9.25. LED Manipulation Commands	86
9.26. Armadillo-200 Series Compatible LED Driver Configuration	87
9.27. Armadillo-400 Series Button Events	87
9.28. Button Configuration	87
9.29. Real-Time Clock I2C Bus Connection	88
9.30. Real-Time Clock sysfs Interface	88
9.31. Real-Time Clock Configuration	88
9.32. Real-Time Clock Selection Configuration	89
9.33. Alarm Interrupt Types	89
9.34. Real-Time Clock Alarm Function Configuration	90
9.35. I2C Configuration	91
9.36. SPI Configuration	92
9.37. One Wire Configuration	93
9.38. PWM sysfs	93
9.39. PWM Configuration	93
9.40. CAN sysfs	94
9.41. CAN Configuration	95
9.42. Keypad Configuration	95
9.43. Sleep States	96
9.44. Wakeup Basis Designation	96
9.45. Wakeup Basis Default Value Configuration	97
9.46. Power Line and Regulator Relationships	97
9.47. Regulators Used By Devices	98
10.1. Armadillo-460 Expansion Bus Operating Mode Configuration	100
10.2. Direct CPU Bus Mode (Synchronous) Configuration	100
10.3. Direct CPU Bus Mode (Asynchronous) Configuration	101
10.4. PC/104 Expansion Bus Compatibility Mode Memory Map	101
10.5. Direct CPU Bus Mode Memory Map	102
10.6. Accessible CS3/CS4 Space In Direct CPU Bus Mode	102
10.7. Interrupt Signals and Corresponding Armadillo-460 Interrupt Numbers	103
10.8. Interrupt Number Conversion Macro Specifications	104
A.1. Well Used Linux Kernel Parameters	109
A.2. frob Command	110
A.3. tftpd Options	111

Chapter 1. Preface

The Armadillo Series are small high-performance low-power general purpose boards which incorporate ARM CPU cores. Linux (kernel 2.6) is employed as the standard operating system, providing access to a rich array of software resources and proven stability. All boards include network interfaces as standard which, combined with the Linux network protocol stack, enable simple development of network ready devices.

The Armadillo-400 Series models provide improved performance over existing products of the same class, while at the same time also offering even lower power consumption. The Armadillo-400 Series is comprised of three products: the low cost Armadillo-420, the Armadillo-440 which can readily support multimedia functionality with the addition of an expansion board, and the Armadillo-460 which includes an expansion bus which conforms with the PC/104 standard.

Armadillo-400 Series boards have interfaces which are often required for embedded devices, such as serial, Ethernet, USB, storage (microSD/SD) and GPIO. In addition to these, multimedia functionality including LCD, touch screen and audio interfaces can be added to Armadillo-440 and Armadillo-460 via an expansion board. Other functionality such as a real-time clock and wireless LAN can also be added with optional modules.

Along with the stand-alone models in the Armadillo-400 Series, there are also models which include expansion boards and option modules and can be used to quickly start prototype development or evaluation. The names and makeup of each model are shown in Table 1.1, “Armadillo-400 Series Models”.

Table 1.1. Armadillo-400 Series Models

Name	Makeup
Armadillo-420 Basic Model	Armadillo-420 + Armadillo-400 Series RTC Option Module
Armadillo-420 WLAN Model (AWL12 Compatible)	Armadillo-420 + Armadillo-400 Series WLAN Option Module (AWL12 Compatible)
Armadillo-420 WLAN Model (AWL13 Compatible)	Armadillo-420 + Armadillo-400 Series WLAN Option Module (AWL13 Compatible)
Armadillo-440 LCD Model	Armadillo-440 + Armadillo-400 Series LCD Expansion Board
Armadillo-460 Basic Model	Armadillo-460

This document provides information required when customizing the software on the Armadillo-400 Series.

The manuals covering default software operation and hardware specifications are different for each model. Each model name and the corresponding manuals are shown in Table 1.2, “Models and Corresponding Manuals”.

Table 1.2. Models and Corresponding Manuals

Name	Software Operation	Hardware Specifications
Armadillo-420 Basic Model	Armadillo-420 Basic Model Development Set Startup Guide	Armadillo-400 Series Hardware Manual
Armadillo-420 WLAN Model (AWL12 Compatible)	Armadillo-420 WLAN Model Development Set Startup Guide	Armadillo-400 Series Hardware Manual
	Armadillo-WLAN Software Manual	Armadillo-WLAN Hardware Manual
Armadillo-420 WLAN Model (AWL13 Compatible)	Armadillo-420 WLAN Model Development Set (AWL13 Compatible) Startup Guide	Armadillo-400 Series Hardware Manual
	Armadillo-WLAN (AWL13) Software Manual	Armadillo-WLAN (AWL13) Hardware Manual
Armadillo-440 LCD Model	Armadillo-440 LCD Model Development Set Startup Guide	Armadillo-400 Series Hardware Manual
Armadillo-460 Basic Model	Armadillo-460 Basic Model Development Set Startup Guide	Armadillo-400 Series Hardware Manual

The product name "Armadillo" will be used in descriptions that apply to the whole Armadillo Series for the remainder of this document.

1.1. Document and Related Files Versions

For all manuals including this document and also all other related files such as source files and image files, we recommend using the newest version available. Before continuing with this document, please check the Armadillo Site (<http://armadillo.atmark-techno.com>) for information on the latest versions.

1.2. Who Should Read This Document

This document is for those planning to customize the software on Armadillo.

1.3. Document Structure

This document comprises of Chapters 1 to 8 and an appendix.

Chapters 1 to 4 deal with the preparation necessary in order to begin development.

Chapters 5 to 7 explain how to set up the development environment, build bootloader, kernel and userland image files from source, and how to write the image files to Armadillo.

Chapter 8 explains how to deploy kernel and userland images to storage devices other than the on-board flash memory.

Chapter 9 describes the specifications of the Linux kernel device drivers unique to Armadillo.

Chapter 10 explains about the expansion bus on Armadillo-460.

Finally, the Appendix explains the functionality of the bootloader.

1.4. Typographical Conventions

1.4.1. Fonts

Fonts are used in the following ways in this document.

Table 1.3. Fonts

Font Example	Description
Plain text font	Used for standard text
[PC ~] \$ ls	Shell prompt and user input text
text	Text that is either displayed, is to be edited, or is a comment

1.4.2. Command Entry Examples

The command entry examples in this document all have an assumed execution environment which is reflected in the displayed prompt. The directory part “/” will differ depending on the current directory. The home directory of each user is represented by “~”.

Table 1.4. Relationship Between Prompt and Execution Environment

Prompt	Command Execution Environment
[PC /] #	To be executed by a privileged user on the work PC
[PC /] \$	To be executed by a general user on the work PC
[armadillo /] #	To be executed by a privileged user on Armadillo
[armadillo /] \$	To be executed by a general user on Armadillo
hermit >	To be executed on Armadillo in maintenance mode

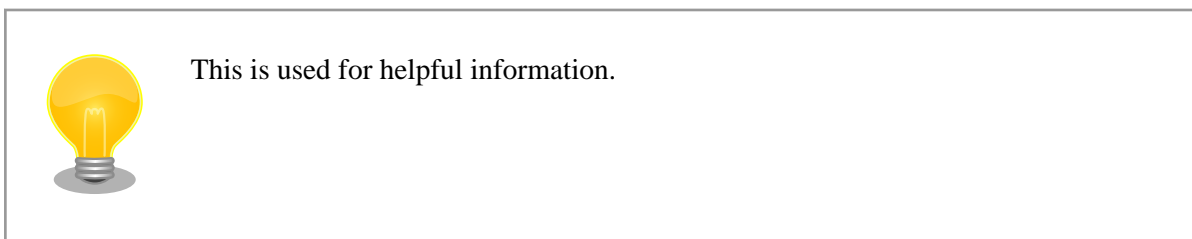
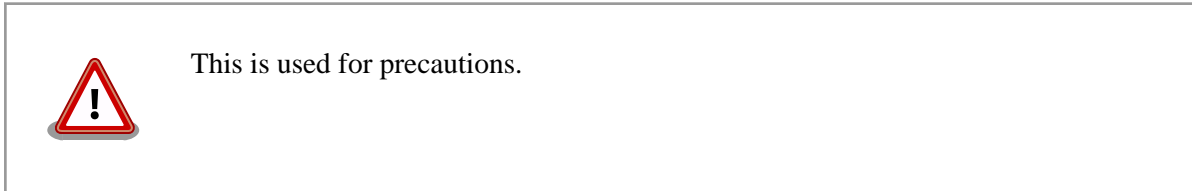
Commands that may change or vary depending on the relevant environment are written as shown below. Please adjust the commands as necessary.

Table 1.5. Abbreviations Used In Command Entry Examples

Notation	Description
[version]	File version number

1.4.3. Icons

Icons are used in the following way in this document.



1.5. Acknowledgements

The software used on Armadillo is composed from Free Software / Open Source Software. This Free Software / Open Source Software is the result of efforts from developers from all over the world. We would like to take this opportunity to express our gratitude.

Chapter 2. Precautions

2.1. Safety Precautions

In order to use this product safely, please take special note of the following precautions.



- Be sure to read all product manuals and related documentation before using this product. Please use this product correctly and safely making sure to follow all usage precautions.
- When operating or extending this product in a way not described in the product manuals, please do so safely and on your own responsibility after having fully understood the materials on our web site and any other technical information.
- Please do not install this product in a place with a lot of water, moisture, dust or soot. This could cause a fire, product failure or electric shock.
- Some parts of this product generate heat and can reach high temperatures. Depending on the surrounding temperature and on how this product is handled, this may cause burns. Please do not touch the electronic components or the surrounding area while the product is powered on or before it has cooled down after being powered off.
- When using this product in the development of devices or systems to original specifications, please carry out the design and development after having thoroughly read and fully understood the product manuals and related materials, the technical information offered on our web site and related device data sheets. Also, please carry out full tests beforehand in order to provide and maintain reliability and safety.
- This product is not intended for uses that require extremely high reliability and safety in terms of functionality and accuracy (such as medical equipment, traffic control systems, combustion control systems, safety equipment and so on). If this product is used in these kinds of equipment, devices or systems, this company will not be held responsible in any way for any accident resulting in injury or death, fire or damage and so on.
- This product uses semiconductor components designed for generic electronics equipment such as office automation equipment, communications equipment, measurement equipment and machine tools. It is possible that a foreign noise or surge may cause this product to malfunction or fail. To ensure there will be no risk to life, the body or property in the event of malfunction or failure, be sure to take all possible measures in regard to device safety design, such as using protection circuits like limit switches or fuse breakers, or system redundancy, and to only use the device after taking measures to ensure sufficient reliability and safety.

- Please do not use products with Wireless LAN functionality in places near medical devices such as heart pacemakers and hearing aids, automatic control equipment such as fire alarms and automatic doors, microwave ovens, advanced electronic equipment or televisions and radios, or near "Premises Radio Stations" for "Mobile Body Identification" or "Specified Low Power Radio Stations". The radio waves emitted by this product may cause these types of devices to malfunction.

2.2. Handling Precautions

Please pay attention to the following points when handling this product in order to avoid causing any irreversible damage.

Areas Easily Damaged	The microSD connector and its cover and the connectors of the flat cable from Armadillo-440 and Armadillo-460 to the LCD Expansion Board can be easily damaged. Please be careful not to damage them by handling them with too much force.
Modifications To This Product	Please take note that any modifications ^[1] made to this product are not covered under warranty. Also, please ensure to undertake a full operational check of this product before carrying out any modifications or mounting connectors ^[2] .
Mounting and Dismounting of Connectors While Powered On	Apart from hot-pluggable interfaces (LAN, USB, SD, Mic, Headphone), do not under any circumstances insert or remove connectors while power is supplied to this product or peripheral circuits.
Static Electricity	As CMOS devices are used in this product, please store it in antistatic packaging (such as that it was shipped in) until time of use.
Latch-up	Excessive noise or a surge from the power supply or input/output, or sharp voltage fluctuations can lead to the CMOS devices incorporated in the board causing a latch-up. Once the latch-up occurs, this situation continues until the power supply is disconnected, and therefore can damage the devices. Measures such as adding a protection circuit to noise-susceptible input/output lines or not sharing the power supply with devices that can be the cause of noise are highly recommended.
Physical Stress	Please avoid strong physical stress such as drops or other impacts.
Touch Panel Operation	The touch panel LCD module on the LCD expansion board is fixed with flexible double sided tape. If a strong force is applied to the LCD screen the double sided tape may give and the LCD frame may touch the board wiring. Please take care not to push the LCD screen stronger than necessary.

2.3. Software Usage Precautions

About Software Contained In This Product	The software and documentation contained in this product is provided “AS IS”. The customer is required to assume the responsibility of only using this product after having fully considered and tested its suitability to the intended purpose and use. There is no guarantee of fitness for a particular purpose, reliability, correctness and no guarantee of any outcomes resulting from the use of this product.
--	---

^[1]With the exception of any methods of modification introduced in this and related product manuals, and the mounting of connectors to unmounted interfaces.
^[2]When making modifications or mounting connectors, please ensure to apply masking and avoid solder residue or solder balls coming in contact with surrounding parts.

2.4. Write Prohibited Regions



The data stored by the EEPROM, CPLD and i.MX257 electrical fuse (e-Fuse) is used by the software contained in this product. Please do not write to these regions as the product may stop operating correctly. Purposefully writing to these regions voids the product warranty.

2.5. Electromagnetic Interference



The Armadillo-400 Series are Class A Information Technology Equipment^[3] as defined under VCCI Council standards. There are cases where this type of equipment can cause electromagnetic interference when used in home environments. In this situation, the user may be required to take appropriate measures to alleviate the problem.



The Armadillo-440 LCD Model (Armadillo-440 together with the Armadillo-400 Series LCD Expansion Board fixed on an acrylic board) does not meet the VCCI standard and can cause electromagnetic interference.

In order to clear Class A when using the Armadillo-400 Series LCD Expansion Board included in the Armadillo-440 LCD Model, it is necessary to strengthen the ground of the expansion board. For example, by using a metal instead of acrylic board or connecting the fixing holes of the Armadillo-440 and the LCD Expansion Board with a wide conducting line.

Please be aware of the following points when newly designing an expansion board which connects to the LCD interface on Armadillo-440 or Armadillo-460.



With an expansion board that includes a device that has large power use fluctuations, such as with an audio amp, when only the GND line of the flexible flat cable (FCC) is connected the expansion board may produce electromagnetic noise. To mitigate the noise, strengthening of the expansion board's ground is recommended. For example, by connecting the fixing holes of the Armadillo-440 or Armadillo-460 and the expansion board GND by metal plate or wide conducting line.

2.6. Warranty

As laid out in the Product Warranty Policy which is provided with this product or available on our web site, the main board of this product is covered by a one year replacement warranty from time of purchase. Please note that the other included goods and software are not covered by the warranty.

^[3]Armadillo-420, Armadillo-440 and Armadillo-460 have cleared Class A when tested with the AC adapter included in the Development Set (UNIFIVE US300520).

Product Warranty Policy <http://www.atmark-techno.com/support/warranty-policy>

2.7. Exporting

This product has as a general rule been developed and manufactured with the assumption that it will be used within Japan. When exporting this product, it is the responsibility of the exporter to follow all export related law and carry out all required procedures. No guarantee is made in regards to whether or not this product conforms to any overseas laws or regulations. This product and related technology may not be used for the purpose of development of weapons of mass destruction, for the purpose of military use or other military related uses, or in devices which have had their production, use, sale or procurement prohibited by national or overseas law or regulations.

2.8. Trademarks

- Armadillo is a registered trademark of Atmark Techno, Inc. All other company names, product names and related trademarks are the property of their respective owners. TM and © marks are omitted.
- The SD, SDHC, microSD, microSDHC and SDIO logos are trademarks of SD-3C, LLC.



Chapter 3. System Overview

This chapter provides a basic system overview.

3.1. Armadillo-400 Series Basic Specifications

The basic specifications of the Armadillo-400 Series boards in their default state^[1] are show in Table 3.1, “Armadillo-400 Series Basic Specifications”. The block diagram for Armadillo-420 and Armadillo-440 is shown in Figure 3.1, “Armadillo-420/440 Block Diagram” and the block diagram for Armadillo-460 is shown in Figure 3.2, “Armadillo-460 Block Diagram”.

Table 3.1. Armadillo-400 Series Basic Specifications

	Armadillo-420	Armadillo-440	Armadillo-460
Processor	Freescale i.MX257 (ARM926EJ-S) Instruction / Data Cache: 16KByte / 16KByte Internal RAM: 128KByte		
System Clock	CPU Core Clock: 400MHz BUS Clock: 133MHz		
RAM	LPDDR SDRAM: 64MByte (16bit width)	LPDDR SDRAM: 128MByte (16bit width)	
ROM	NOR Flash Memory: 16MByte (16bit width)	NOR Flash Memory: 32MByte (16bit width)	
Serial	RS232C Levels x1 port Flow control pins (full modem) 230.4 kbps max		RS232C Levels x2 ports Flow control pins (full modem 1 port) 230.4 kbps max
	3.3V I/O levels x2 ports No flow control pins 4Mbps max		
USB 2.0 Host	High Speed x1 port		
	Full Speed x1 port		
LAN	10BASE-T/100BASE-TX x1 port		
Storage	microSD x1 4bit width, 208Mbps max		SD x1 4bit width, 208Mbps max
	GPIO 3.3V I/O levels x18 pins		
Programmable LEDs	Red x1, Green x1, Yellow x1		
Buttons	Tact switch x1		Tact switch x1
			Reset switch x1

^[1]It is possible to change the functions assigned to the I/O pins on the Armadillo-400 Series with multiplexing. For more information please refer to "Armadillo-400 Series Hardware Manual" and Chapter 9, Linux Kernel Device Driver Specifications.

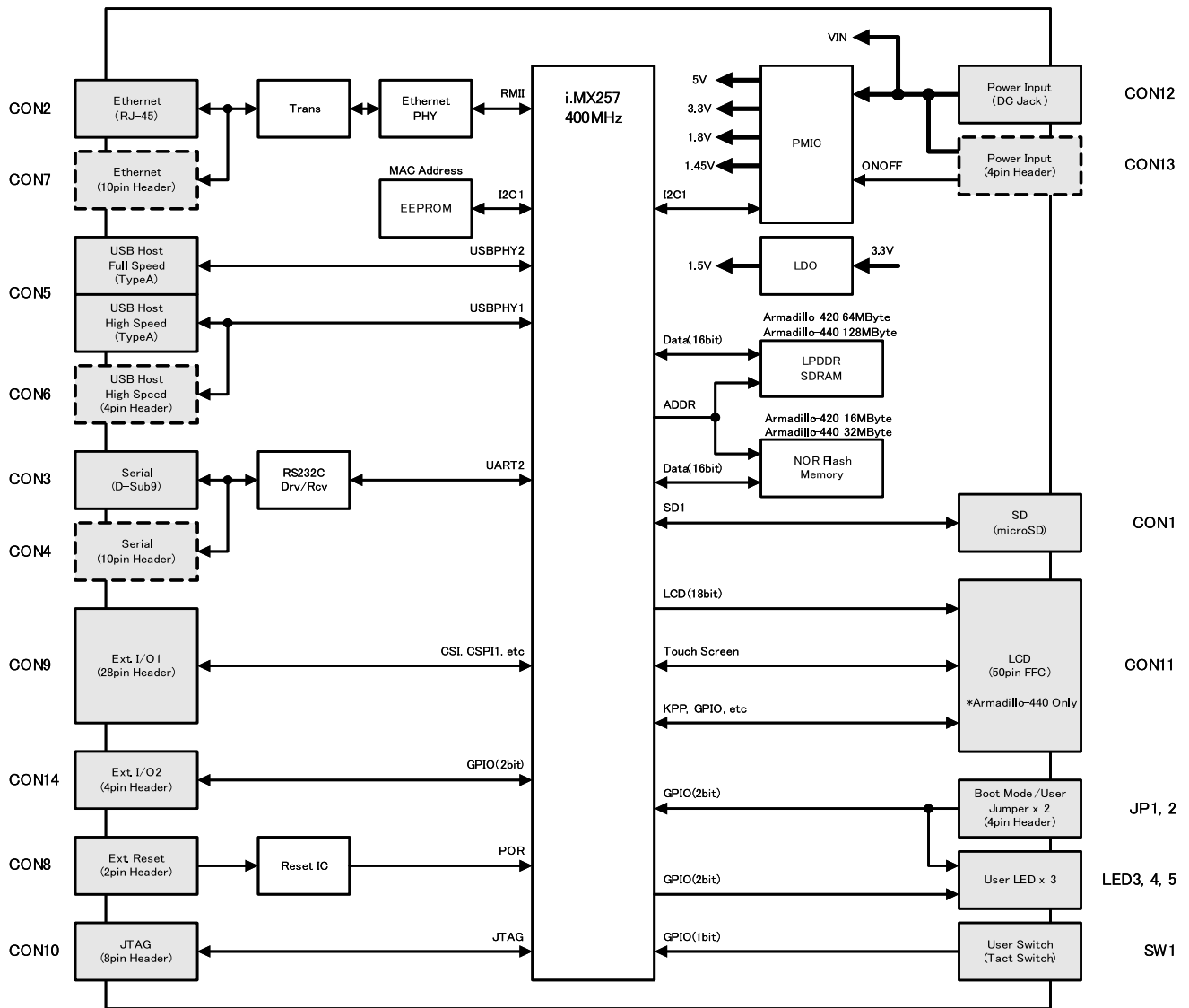


Figure 3.1. Armadillo-420/440 Block Diagram

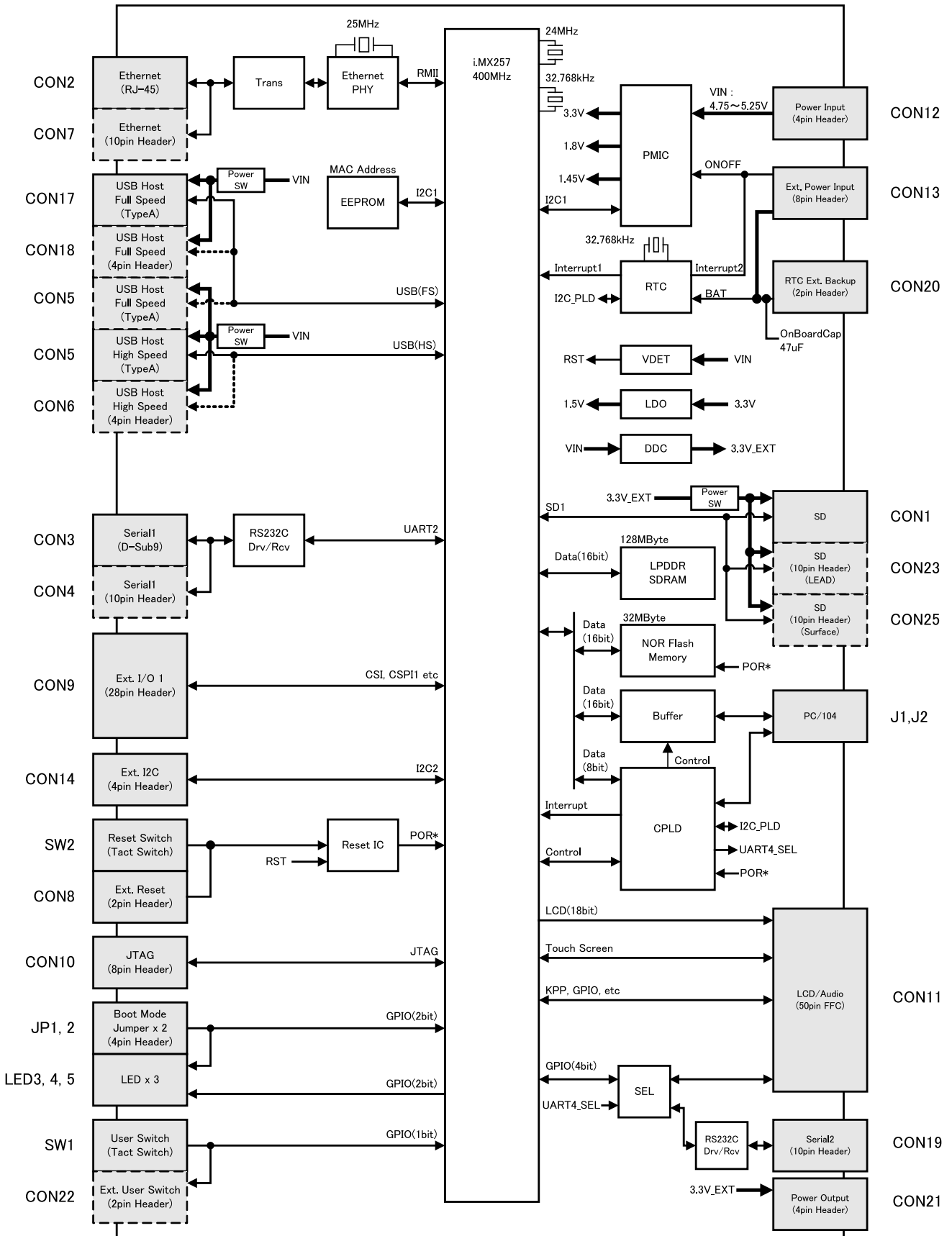


Figure 3.2. Armadillo-460 Block Diagram

3.2. Basic Specifications of Armadillo-420 Basic Model

The Armadillo-420 Basic Model is comprised of the Armadillo-420 board together with the Armadillo-400 Series RTC Option Module. The basic specifications of the RTC Option Module are shown in Table 3.2, “Basic Specifications of RTC Option Module”.

Table 3.2. Basic Specifications of RTC Option Module

Armadillo-400 Series RTC Option Module	
Real-time Clock	Operation possible for a certain period of time after power off



The backup time of the real-time clock differs depending on the RTC Option Module product number. Also, it is possible to keep time data even after power has been cut for an extended period of time by connecting an external battery. For detailed specifications, please refer to the "Armadillo-400 Series Hardware Manual".

The basic layout of the Armadillo-420 Basic Model is shown in Figure 3.3, “Basic Layout of Armadillo-420 Basic Model”. The pin states of the expansion interfaces (CON9 and CON14) after the Linux kernel has booted when using the default image are shown in Table 3.3, “Default States of Armadillo-420 Basic Model Expansion Interfaces”^[2]. Please make sure to confirm the position of each interface.

^[2]For the states before the Linux kernel has booted, please refer to the "Armadillo-400 Series Hardware Manual".

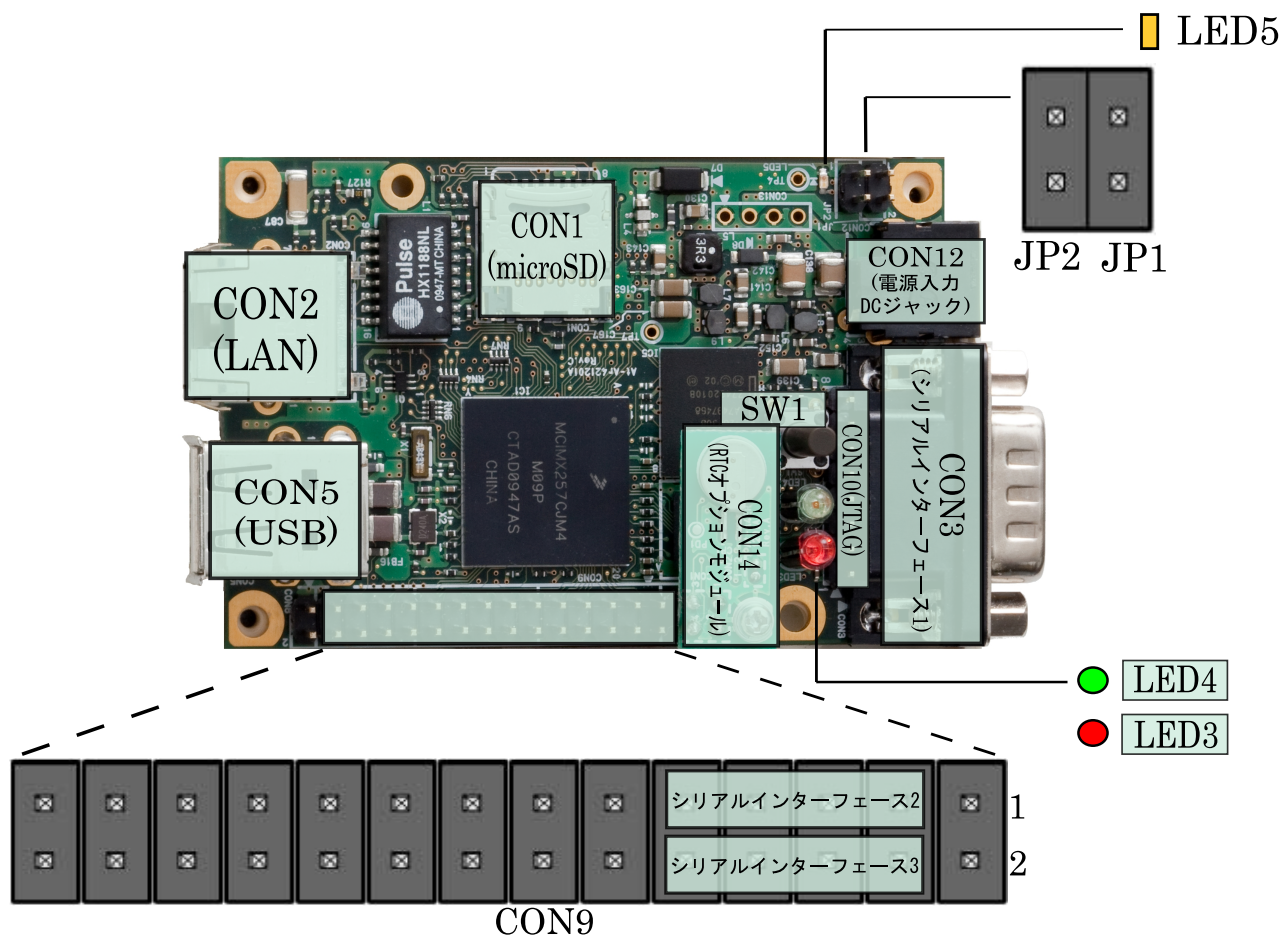


Figure 3.3. Basic Layout of Armadillo-420 Basic Model

Table 3.3. Default States of Armadillo-420 Basic Model Expansion Interfaces

Pin Number	Function	Input/Output	Open Drain	Pull/Keeper ^[a]	Slew Rate	Drive Strength
CON14 1 ^[b]	+3.3V_IO	Power	-	-	-	-
CON14 2	GND	Power	-	-	-	-
CON14 3	I2C2_SCL	Input/Output	Enabled	22kΩ PU ^[c]	Slow	Std.
CON14 4	I2C2_SDA	Input/Output	Enabled	22kΩ PU ^[c]	Slow	Std.
CON9 1	GPIO3_17	Input	Disabled	100kΩ PU	Slow	Std.
CON9 2	GPIO3_14	Input	Disabled	100kΩ PU	Slow	Std.
CON9 3	Serial Interface 2 UART3_RXD	Input	Disabled	100kΩ PU	Slow	Std.
CON9 4	Serial Interface 3 UART5_RXD	Input	Disabled	100kΩ PU	Slow	Std.
CON9 5	Serial Interface 2 UART3_TXD	Output	Disabled	Disabled	Slow	Std.
CON9 6	Serial Interface 3 UART5_TXD	Output	Disabled	Disabled	Slow	Std.
CON9 7	+3.3V_IO	Power	-	-	-	-
CON9 8	+3.3V_IO	Power	-	-	-	-
CON9 9	GND	Power	-	-	-	-
CON9 10	GND	Power	-	-	-	-
CON9 11	GPIO1_17	Input	Disabled	100kΩ PU	Slow	Std.
CON9 12	GPIO1_29	Input	Disabled	100kΩ PU	Slow	Std.
CON9 13	GPIO1_18	Input	Disabled	100kΩ PU	Slow	Std.

Pin Number	Function	Input/Output	Open Drain	Pull/Keeper ^[a]	Slew Rate	Drive Strength
CON9 14	GPIO1_30	Input	Disabled	100kΩ PU	Slow	Std.
CON9 15	GPIO1_7	Input	Disabled	100kΩ PU	Slow	Std.
CON9 16	GPIO1_31	Input	Disabled	100kΩ PU	Slow	Std.
CON9 17	GPIO4_21	Input	Disabled	100kΩ PU	Slow	Std.
CON9 18	GPIO1_6	Input	Disabled	100kΩ PU	Slow	Std.
CON9 19	GND	Power	-	-	-	-
CON9 20	+3.3V_IO	Power	-	-	-	-
CON9 21	GPIO1_8	Input	Disabled	100kΩ PU	Slow	Std.
CON9 22	GPIO1_9	Input	Disabled	100kΩ PU	Slow	Std.
CON9 23	GPIO1_10	Input	Disabled	100kΩ PU	Slow	Std.
CON9 24	GPIO1_11	Input	Disabled	100kΩ PU	Slow	Std.
CON9 25	GPIO1_16	Input	Disabled	100kΩ PU	Slow	Std.
CON9 26	GPIO2_22	Input	Disabled	100kΩ PU	Slow	Std.
CON9 27	GPIO2_21	Output Low	Disabled	Disabled	Fast	Std.
CON9 28	GPIO3_15	Output Low	Disabled	Disabled	Fast	Std.

^[a]PD=pull-down, PU=pull-up.

^[b]Connected to the CON1 1 pin on the RTC Option Module. Below to CON14 4 connected to the RTC Option Module.

^[c]1kΩ PU on the RTC Option Module.



Serial Interfaces 2 and 3 have +3.3V I/O levels. They can be used at RS232C levels by connecting the optional^[3] RS232C level conversion adapter.

When using the RS232C level conversion adapter, please connect pin 1 (the yellow or green wire) to CON9 1 for Serial Interface 2 and to CON9 2 for Serial Interface 3.

3.3. Basic Specifications of Armadillo-420 WLAN Model (AWL12 Compatible)

The Armadillo-420 WLAN Model (AWL12 Compatible) is a model with the Armadillo-400 Series WLAN Option Module (AWL12 Compatible) (hereafter WLAN Option Module (AWL12 Compatible)) connected to Armadillo-420. The basic specifications of the WLAN Option Module (AWL12 Compatible) are shown in Table 3.4, “Basic Specifications of WLAN Option Module (AWL12 Compatible)”.

Table 3.4. Basic Specifications of WLAN Option Module (AWL12 Compatible)

	Armadillo-400 Series WLAN Option Module (AWL12 Compatible)
Wireless LAN Specifications	IEEE802.11b, IEEE802.11g, IEEE802.11i
Transceiver Frequencies	2400MHz - 2483.5MHz (ch1 - 13)
Access Methods	Infrastructure mode, ad-hoc mode
Security Methods	64bit/128bit WEP, TKIP, AES
Real-time Clock	Operation possible for a certain period of time after power off



With the real-time clock, it is possible to keep time data even after power has been cut for an extended period of time by connecting an external battery. For detailed specifications, please refer to the "Armadillo-400 Series Hardware Manual".

^[3]A RS232C level conversion adapter is available as an option and is also included in the development set.

The basic layout of the Armadillo-420 WLAN Model (AWL12 Compatible) is shown in Figure 3.4, “Basic Layout of Armadillo-420 WLAN Model (AWL12 Compatible)”. The pin states of the expansion interfaces (CON9 and CON14) after the Linux kernel has booted when using the default image are shown in Table 3.5, “Default States of Armadillo-420 WLAN Model (AWL12 Compatible) Expansion Interfaces”^[4]. Please make sure to confirm the position of each interface.

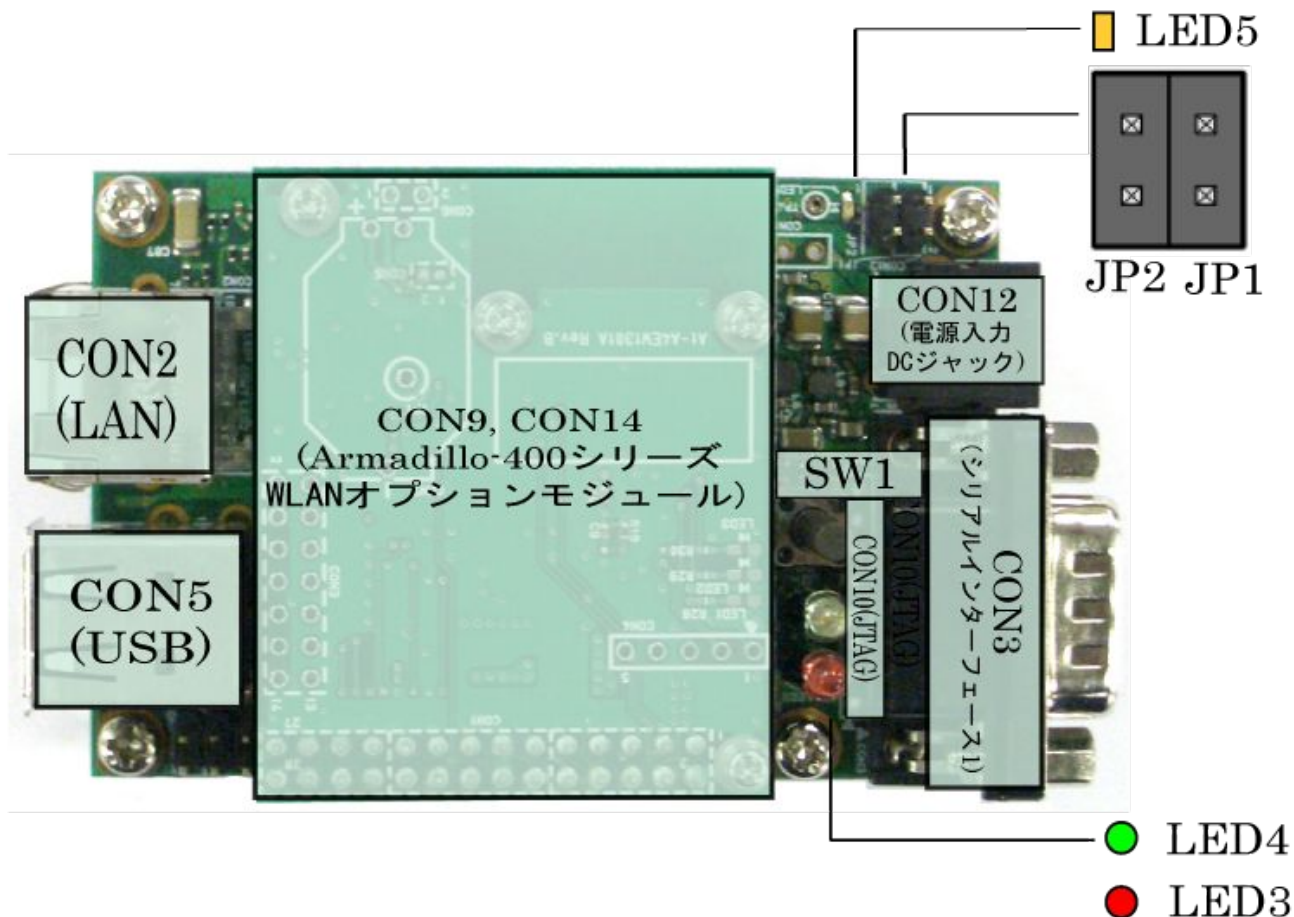


Figure 3.4. Basic Layout of Armadillo-420 WLAN Model (AWL12 Compatible)

Table 3.5. Default States of Armadillo-420 WLAN Model (AWL12 Compatible) Expansion Interfaces

Pin Number	Function	Input/Output	Open Drain	Pull/Keeper ^[a]	Slew Rate	Drive Strength
CON14 1 ^[b]	+3.3V_IO	Power	-	-	-	-
CON14 2	GND	Power	-	-	-	-
CON14 3	I2C2_SCL	Input/Output	Enabled	22kΩ PU ^[c]	Slow	Std.
CON14 4	I2C2_SDA	Input/Output	Enabled	22kΩ PU ^[c]	Slow	Std.
CON9 1	SDHC2_PWREN (GPIO3_17)	Output	Disabled	100kΩ PU ^[d]	Slow	Std.
CON9 2	RTC_INT1 (GPIO3_14)	Input	Disabled	22kΩ PU	Slow	Std.
CON9 3	GPIO1_14 ^[e]	Input	Disabled	100kΩ PU	Slow	Std.
CON9 4	Serial Interface 3 UART5_RXD	Input	Disabled	100kΩ PU	Slow	Std.
CON9 5	GPIO1_15 ^[e]	Input	Disabled	100kΩ PU	Slow	Std.
CON9 6	Serial Interface 3 UART5_TXD	Output	Disabled	Disabled	Slow	Std.
CON9 7	+3.3V_IO	Power	-	-	-	-

^[4]For the states before the Linux kernel has booted, please refer to the "Armadillo-400 Series Hardware Manual".

Pin Number	Function	Input/Output	Open Drain	Pull/Keeper ^[a]	Slew Rate	Drive Strength
CON9 8	+3.3V_IO	Power	-	-	-	-
CON9 9	GND	Power	-	-	-	-
CON9 10	GND	Power	-	-	-	-
CON9 11	GPIO1_17 ^[e]	Input	Disabled	100kΩ PU	Slow	Std.
CON9 12	GPIO1_29 ^[e]	Input	Disabled	100kΩ PU	Slow	High
CON9 13	GPIO1_18 ^[e]	Input	Disabled	100kΩ PU	Slow	Std.
CON9 14	GPIO1_30 ^[e]	Input	Disabled	100kΩ PU	Slow	High
CON9 15	SDHC2_WP (GPIO1_7)	Input	Disabled	100kΩ PU ^[f]	Slow	High
CON9 16	SDHC2_CMD	Input/Output	Disabled	Disabled ^[g]	Fast	High
CON9 17	SDHC2_CD (GPIO4_21)	Input	Disabled	100kΩ PU ^[f]	Slow	High
CON9 18	SDHC2_CLK	Output	Disabled	Disabled	Fast	High
CON9 19	GND	Power	-	-	-	-
CON9 20	+3.3V_IO	Power	-	-	-	-
CON9 21	SDHC2_DATA0	Input/Output	Disabled	Disabled ^[g]	Fast	High
CON9 22	SDHC2_DATA1	Input/Output	Disabled	Disabled ^[g]	Fast	High
CON9 23	SDHC2_DATA2	Input/Output	Disabled	Disabled ^[g]	Fast	High
CON9 24	SDHC2_DATA3	Input/Output	Disabled	Disabled ^[g]	Fast	High
CON9 25	GPIO1_16	Input	Disabled	100kΩ PU	Slow	Std.
CON9 26	GPIO2_22	Input	Disabled	100kΩ PU	Slow	Std.
CON9 27	GPIO2_21	Output Low	Disabled	Disabled	Fast	Std.
CON9 28	GPIO3_15	Output Low	Disabled	Disabled	Fast	Std.

^[a]PD=pull-down, PU=pull-up

^[b]Connected to the CON1 1 pin on WLAN Option Module (AWL12 Compatible). Below to CON9 24, connected to LAN Option Module (AWL12 Compatible).


^[c]1kΩ PU on the WLAN Option Module (AWL12 Compatible).

^[d]1kΩ PD on the WLAN Option Module (AWL12 Compatible).

^[e]Unused on the WLAN Option Module (AWL12 Compatible).

^[f]10kΩ PD on the WLAN Option Module (AWL12 Compatible).

^[g]47kΩ PU on the WLAN Option Module (AWL12 Compatible).



As the CON9 2 pin did not support the RTC alarm interrupt in linux-2.6.26-at12 (linux-a400-wlan-1.05.bin.gz) it was set to GPIO by default. This changed in linux-2.6.26-at13 (linux-a400-wlan-1.06.bin.gz) and is now set to use RTC INT by default. Please be aware of this change in default settings when using the CON9 2 pin.


3.4. Basic Specifications of Armadillo-420 WLAN Model (AWL13 Compatible)

The Armadillo-420 WLAN Model (AWL13 Compatible) is a model with the Armadillo-400 Series WLAN Option Module (AWL13 Compatible) (hereafter WLAN Option Module (AWL13 Compatible)) connected to Armadillo-420. The basic specifications of the WLAN Option Module (AWL13 Compatible) are shown in Table 3.6, “Basic Specifications of WLAN Option Module (AWL13 Compatible)”.

Table 3.6. Basic Specifications of WLAN Option Module (AWL13 Compatible)

	Armadillo-400 Series WLAN Option Module (AWL13 Compatible)
Wireless LAN Specifications	IEEE802.11b, IEEE802.11g, IEEE802.11n, IEEE802.11i
Transceiver Frequencies	2400MHz - 2483.5MHz (ch1 - 13)
Access Methods	Infrastructure mode, ad-hoc mode

	Armadillo-400 Series WLAN Option Module (AWL13 Compatible)
Security Methods	64bit/128bit WEP, TKIP, AES
Real-time Clock	Operation possible for a certain period of time after power off



With the real-time clock, it is possible to keep time data even after power has been cut for an extended period of time by connecting an external battery. For detailed specifications, please refer to the "Armadillo-400 Series Hardware Manual".

The basic layout of the Armadillo-420 WLAN Model (AWL13 Compatible) is shown in Figure 3.5, “Basic Layout of Armadillo-420 WLAN Model (AWL13 Compatible)”. The pin states of the expansion interfaces (CON9 and CON14) after the Linux kernel has booted when using the default image are shown in Table 3.7, “Default States of Armadillo-420 WLAN Model (AWL13 Compatible) Expansion Interfaces”^[5]. Please make sure to confirm the position of each interface.

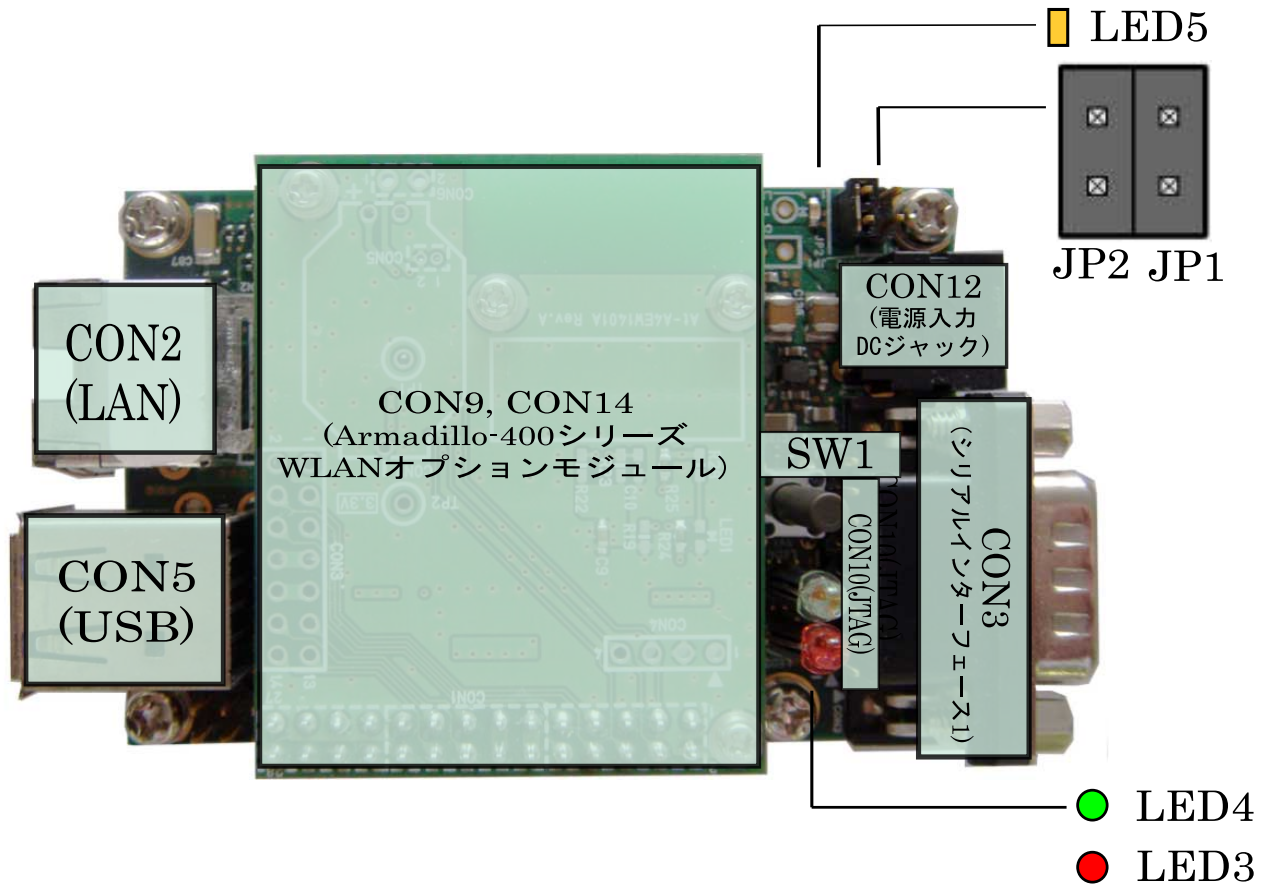


Figure 3.5. Basic Layout of Armadillo-420 WLAN Model (AWL13 Compatible)

Table 3.7. Default States of Armadillo-420 WLAN Model (AWL13 Compatible) Expansion Interfaces

Pin Number	Function	Input/Output	Open Drain	Pull/Keeper ^[a]	Slew Rate	Drive Strength
CON14 1 ^[b]	+3.3V_IO	Power	-	-	-	-
CON14 2	GND	Power	-	-	-	-

^[5]For the states before the Linux kernel has booted, please refer to the "Armadillo-400 Series Hardware Manual".

Pin Number	Function	Input/Output	Open Drain	Pull/Keeper ^[a]	Slew Rate	Drive Strength
CON14 3	I2C2_SCL	Input/Output	Enabled	22kΩ PU ^[c]	Slow	Std.
CON14 4	I2C2_SDA	Input/Output	Enabled	22kΩ PU ^[c]	Slow	Std.
CON9 1	SDHC2_PWREN (GPIO3_17)	Output	Disabled	100kΩ PU ^[d]	Slow	Std.
CON9 2	RTC_INT1 (GPIO3_14)	Input	Disabled	22kΩ PU	Slow	Std.
CON9 3	GPIO1_14 ^[e]	Input	Disabled	100kΩ PU	Slow	Std.
CON9 4	Serial Interface 3 UART5_RXD	Input	Disabled	100kΩ PU	Slow	Std.
CON9 5	GPIO1_15 ^[e]	Input	Disabled	100kΩ PU	Slow	Std.
CON9 6	Serial Interface 3 UART5_TXD	Output	Disabled	Disabled	Slow	Std.
CON9 7	+3.3V_IO	Power	-	-	-	-
CON9 8	+3.3V_IO	Power	-	-	-	-
CON9 9	GND	Power	-	-	-	-
CON9 10	GND	Power	-	-	-	-
CON9 11	GPIO1_17 ^[e]	Input	Disabled	100kΩ PU	Slow	Std.
CON9 12	GPIO1_29 ^[e]	Input	Disabled	100kΩ PU	Slow	High
CON9 13	GPIO1_18 ^[e]	Input	Disabled	100kΩ PU	Slow	Std.
CON9 14	GPIO1_30 ^[e]	Input	Disabled	100kΩ PU	Slow	High
CON9 15	SDHC2_WP (GPIO1_7)	Input	Disabled	100kΩ PU ^[f]	Slow	High
CON9 16	SDHC2_CMD	Input/Output	Disabled	Disabled ^[g]	Fast	High
CON9 17	SDHC2_CD (GPIO4_21)	Input	Disabled	100kΩ PU ^[f]	Slow	High
CON9 18	SDHC2_CLK	Output	Disabled	Disabled	Fast	High
CON9 19	GND	Power	-	-	-	-
CON9 20	+3.3V_IO	Power	-	-	-	-
CON9 21	SDHC2_DATA0	Input/Output	Disabled	Disabled ^[g]	Fast	High
CON9 22	SDHC2_DATA1	Input/Output	Disabled	Disabled ^[g]	Fast	High
CON9 23	SDHC2_DATA2	Input/Output	Disabled	Disabled ^[g]	Fast	High
CON9 24	SDHC2_DATA3	Input/Output	Disabled	Disabled ^[g]	Fast	High
CON9 25	GPIO1_16	Input	Disabled	100kΩ PU	Slow	Std.
CON9 26	GPIO2_22	Input	Disabled	100kΩ PU	Slow	Std.
CON9 27	GPIO2_21	Output Low	Disabled	Disabled	Fast	Std.
CON9 28	GPIO3_15	Output Low	Disabled	Disabled	Fast	Std.

^[a]PD=pull-down, PU=pull-up

^[b]Connected to the CON1 1 pin on WLAN Option Module (AWL13 Compatible). Below to CON9 24, connected to LAN Option Module (AWL13 Compatible).

^[c]1kΩ PU on the WLAN Option Module (AWL13 Compatible).

^[d]1kΩ PD on the WLAN Option Module (AWL13 Compatible).

^[e]Unused on the WLAN Option Module (AWL13 Compatible).

^[f]10kΩ PD on the WLAN Option Module (AWL13 Compatible).

^[g]47kΩ PU on the WLAN Option Module (AWL13 Compatible).



As the CON9 2 pin did not support the RTC alarm interrupt in linux-2.6.26-at12 (linux-a400-wlan-1.05.bin.gz) it was set to GPIO by default. This changed in linux-2.6.26-at13 (linux-a400-wlan-1.06.bin.gz) and is now set to use RTC INT by default. Please be aware of this change in default settings when using the CON9 2 pin.

3.5. Basic Specifications of Armadillo-440 LCD Model

The Armadillo-440 LCD Model is comprised of the Armadillo-440 board together with the Armadillo-400 Series LCD Expansion Board. The basic specifications of the LCD Expansion Board are shown in Table 3.8, “Basic Specifications of Expansion Board”.

Table 3.8. Basic Specifications of Expansion Board

	Armadillo-400 Series LCD Expansion Board
Audio	Playback (stereo) / Capture (mono)
LCD	Resolution: 480 x 272 pixels RGB 565 Color
Touchscreen	4-Wire Resistive
Real-time Clock	Operation possible for a certain period of time after power off
Buttons	Tact switch x3



The backup time of the real-time clock differs depending on the LCD Expansion Board product revision. Also, it is possible to keep time data even after power has been cut for an extended period of time by connecting an external battery. For detailed specifications, please refer to the "Armadillo-400 Series Hardware Manual".

The basic layout of the Armadillo-440 LCD Model is shown in Figure 3.6, “Basic Layout of Armadillo-440 LCD Model”. The pin states of the expansion interfaces (CON9, CON11 and CON14) after the Linux kernel has booted when using the default image are shown in Table 3.9, “Default States of Armadillo-440 LCD Model Expansion Interfaces”^[6]. Please make sure to confirm the position of each interface.

^[6]For the states before the Linux kernel has booted, please refer to the "Armadillo-400 Series Hardware Manual".



Figure 3.6. Basic Layout of Armadillo-440 LCD Model

Table 3.9. Default States of Armadillo-440 LCD Model Expansion Interfaces

Pin Number	Function	Input/Output	Open Drain	Pull/Keeper ^[a]	Slew Rate	Drive Strength
CON14 1	+3.3V_IO	Power	-	-	-	-
CON14 2	GND	Power	-	-	-	-
CON14 3	I2C2_SCL	Input/Output	Enabled	22kΩ PU	Slow	Std.
CON14 4	I2C2_SDA	Input/Output	Enabled	22kΩ PU	Slow	Std.
CON9 1	GPIO3_17	Input	Disabled	100kΩ PU	Slow	Std.
CON9 2	GPIO3_14	Input	Disabled	100kΩ PU	Slow	Std.
CON9 3	Serial Interface 2 UART3_RXD	Input	Disabled	100kΩ PU	Slow	Std.
CON9 4	Serial Interface 3 UART5_RXD	Input	Disabled	100kΩ PU	Slow	Std.
CON9 5	Serial Interface 2 UART3_TXD	Output	Disabled	Disabled	Slow	Std.
CON9 6	Serial Interface 3 UART5_TXD	Output	Disabled	Disabled	Slow	Std.
CON9 7	+3.3V_IO	Power	-	-	-	-
CON9 8	+3.3V_IO	Power	-	-	-	-
CON9 9	GND	Power	-	-	-	-
CON9 10	GND	Power	-	-	-	-
CON9 11	GPIO1_17	Input	Disabled	100kΩ PU	Slow	Std.
CON9 12	GPIO1_29	Input	Disabled	100kΩ PU	Slow	Std.
CON9 13	GPIO1_18	Input	Disabled	100kΩ PU	Slow	Std.
CON9 14	GPIO1_30	Input	Disabled	100kΩ PU	Slow	Std.
CON9 15	GPIO1_7	Input	Disabled	100kΩ PU	Slow	Std.
CON9 16	GPIO1_31	Input	Disabled	100kΩ PU	Slow	Std.
CON9 17	GPIO4_21	Input	Disabled	100kΩ PU	Slow	Std.
CON9 18	GPIO1_6	Input	Disabled	100kΩ PU	Slow	Std.
CON9 19	GND	Power	-	-	-	-
CON9 20	+3.3V_IO	Power	-	-	-	-
CON9 21	GPIO1_8	Input	Disabled	100kΩ PU	Slow	Std.
CON9 22	GPIO1_9	Input	Disabled	100kΩ PU	Slow	Std.
CON9 23	GPIO1_10	Input	Disabled	100kΩ PU	Slow	Std.
CON9 24	GPIO1_11	Input	Disabled	100kΩ PU	Slow	Std.
CON9 25	GPIO1_16	Input	Disabled	100kΩ PU	Slow	Std.
CON9 26	GPIO2_22	Input	Disabled	100kΩ PU	Slow	Std.
CON9 27	GPIO2_21	Output Low	Disabled	Disabled	Fast	Std.
CON9 28	GPIO3_15	Output Low	Disabled	Disabled	Fast	Std.
CON11 1 ^[b]	VIN	Power	-	-	-	-
CON11 2	VIN	Power	-	-	-	-
CON11 3	VIN	Power	-	-	-	-
CON11 4	+3.3V_IO	Power	-	-	-	-
CON11 5	+3.3V_IO	Power	-	-	-	-
CON11 6	GND	Power	-	-	-	-
CON11 7	GND	Power	-	-	-	-
CON11 8	LCDC_LSCLK	Output	Disabled	Disabled ^[c]	Fast	Std.
CON11 9	LCDC_HSYNC	Output	Disabled	Disabled ^[c]	Slow	Std.
CON11 10	LCDC_VSYNC	Output	Disabled	Disabled ^[c]	Slow	Std.
CON11 11	LCDC_OE_ACD	Output	Disabled	Disabled ^[c]	Slow	Std.
CON11 12	PWM1_PWM0	Output	Disabled	100kΩ PU ^{[c][d]}	Slow	Std.
CON11 13	LCDC_LD0	Output	Disabled	Disabled ^[c]	Slow	Std.
CON11 14	LCDC_LD1	Output	Disabled	Disabled ^[c]	Slow	Std.
CON11 15	LCDC_LD2	Output	Disabled	Disabled ^[c]	Slow	Std.
CON11 16	LCDC_LD3	Output	Disabled	Disabled ^[c]	Slow	Std.

Pin Number	Function	Input/Output	Open Drain	Pull/Keeper ^[a]	Slew Rate	Drive Strength
CON11 17	LCDC_LD4	Output	Disabled	Disabled ^[c]	Slow	Std.
CON11 18	LCDC_LD5	Output	Disabled	Disabled ^[c]	Slow	Std.
CON11 19	GND	Power	-	-	-	-
CON11 20	LCDC_LD6	Output	Disabled	Disabled ^[e]	Slow	Std.
CON11 21	LCDC_LD7	Output	Disabled	Disabled ^[c]	Slow	Std.
CON11 22	LCDC_LD8	Output	Disabled	Disabled ^[c]	Slow	Std.
CON11 23	LCDC_LD9	Output	Disabled	Disabled ^[c]	Slow	Std.
CON11 24	LCDC_LD10	Output	Disabled	Disabled ^[c]	Slow	Std.
CON11 25	LCDC_LD11	Output	Disabled	Disabled ^[c]	Slow	Std.
CON11 26	GND	Power	-	-	-	-
CON11 27	LCDC_LD12	Output	Disabled	Disabled ^[c]	Slow	Std.
CON11 28	LCDC_LD13	Output	Disabled	Disabled ^[c]	Slow	Std.
CON11 29	LCDC_LD14	Output	Disabled	Disabled ^[e]	Slow	Std.
CON11 30	LCDC_LD15	Output	Disabled	Disabled ^[e]	Slow	Std.
CON11 31	LCDC_LD16	Output	Disabled	Disabled	Slow	Std.
CON11 32	LCDC_LD17	Output	Disabled	Disabled	Slow	Std.
CON11 33	GND	Power	-	-	-	-
CON11 34	ADC_XP	Analog Input	-	-	-	-
CON11 35	ADC_XN	Analog Input	-	-	-	-
CON11 36	ADC_YP	Analog Input	-	-	-	-
CON11 37	ADC_YN	Analog Input	-	-	-	-
CON11 38	GND	Power	-	-	-	-
CON11 39	LCD_SW1 (GPIO2_20)	Input	Disabled	47kΩ PU	Slow	Std.
CON11 40	LCD_SW2 (GPIO2_29)	Input	Disabled	47kΩ PU	Slow	Std.
CON11 41	LCD_SW3 (GPIO2_30)	Input	Disabled	47kΩ PU	Slow	Std.
CON11 42	AUD5_RXC	Output	Disabled	100kΩ PU	Slow	Std.
CON11 43	AUD5_RXFS ^[f]	Input	Disabled	100kΩ PU	Slow	Std.
CON11 44	AUD5_TXD	Output	Disabled	100kΩ PU	Slow	Std.
CON11 45	AUD5_RXD	Input	Disabled	100kΩ PU	Slow	Std.
CON11 46	AUD5_TXC	Output	Disabled	100kΩ PU	Slow	Std.
CON11 47	AUD5_TXFS	Output	Disabled	100kΩ PU	Slow	Std.
CON11 48	I2C3_SCL	Input/Output	Enabled	22kΩ PU ^[d]	Slow	Std.
CON11 49	I2C3_SDA	Input/Output	Enabled	22kΩ PU ^[d]	Slow	Std.
CON11 50	GND	Power	-	-	-	-

^[a]PD=pull-down, PU=pull-up.

^[b]Connected to CON1 50 on the LCD Expansion Board. Below to CON11 50 connected to the LCD Expansion Board.

^[c]47kΩ PD on the Armadillo-440 board.

^[d]1kΩ PU on the LCD Expansion Board.

^[e]47kΩ PU on the Armadillo-440 board.

^[f]Unused on the LCD Expansion Board.



Serial Interfaces 2 and 3 have +3.3V I/O levels. They can be used at RS232C levels by connecting the optional^[7] RS232C level conversion adapter.

When using the RS232C level conversion adapter, please connect pin 1 (the yellow or green wire) to CON9 1 for Serial Interface 2 and to CON9 2 for Serial Interface 3.

^[7]A RS232C level conversion adapter is available as an option and is also included in the development set.



CON14 3 and CON14 4 were configured to be used as GPIO by default in linux-2.6.26-at7 (linux-a400-1.00.bin.gz). In linux-2.6.26-at8 (linux-a400-1.01.bin.gz) and later, this was changed so that they are configured to be used as I2C2 by default. Please be aware of this change when using CON14 3 and CON14 4.

3.6. Basic Specifications of Armadillo-460 Basic Model

The Armadillo-460 basic model is a Armadillo-460 standalone model.

The basic layout of the Armadillo-460 Basic Model is shown in Figure 3.7, “Basic Layout of Armadillo-460 Basic Model”. The pin states of the expansion interfaces (CON9, CON11 and CON14) after the Linux kernel has booted when using the default image are shown in Table 3.10, “Default States of Armadillo-460 Basic Model Expansion Interfaces”^[8]. Please make sure to confirm the position of each interface. With the default image the Expansion Bus Interface (J1 and J2) is set to PC/104 Expansion Bus Compatibility Mode. For the signal layout of the Expansion Bus Interface in PC/104 Expansion Bus Compatibility Mode, please refer to the "Armadillo-400 Series Hardware Manual".

^[8]For the states before the Linux kernel has booted, please refer to the "Armadillo-400 Series Hardware Manual".

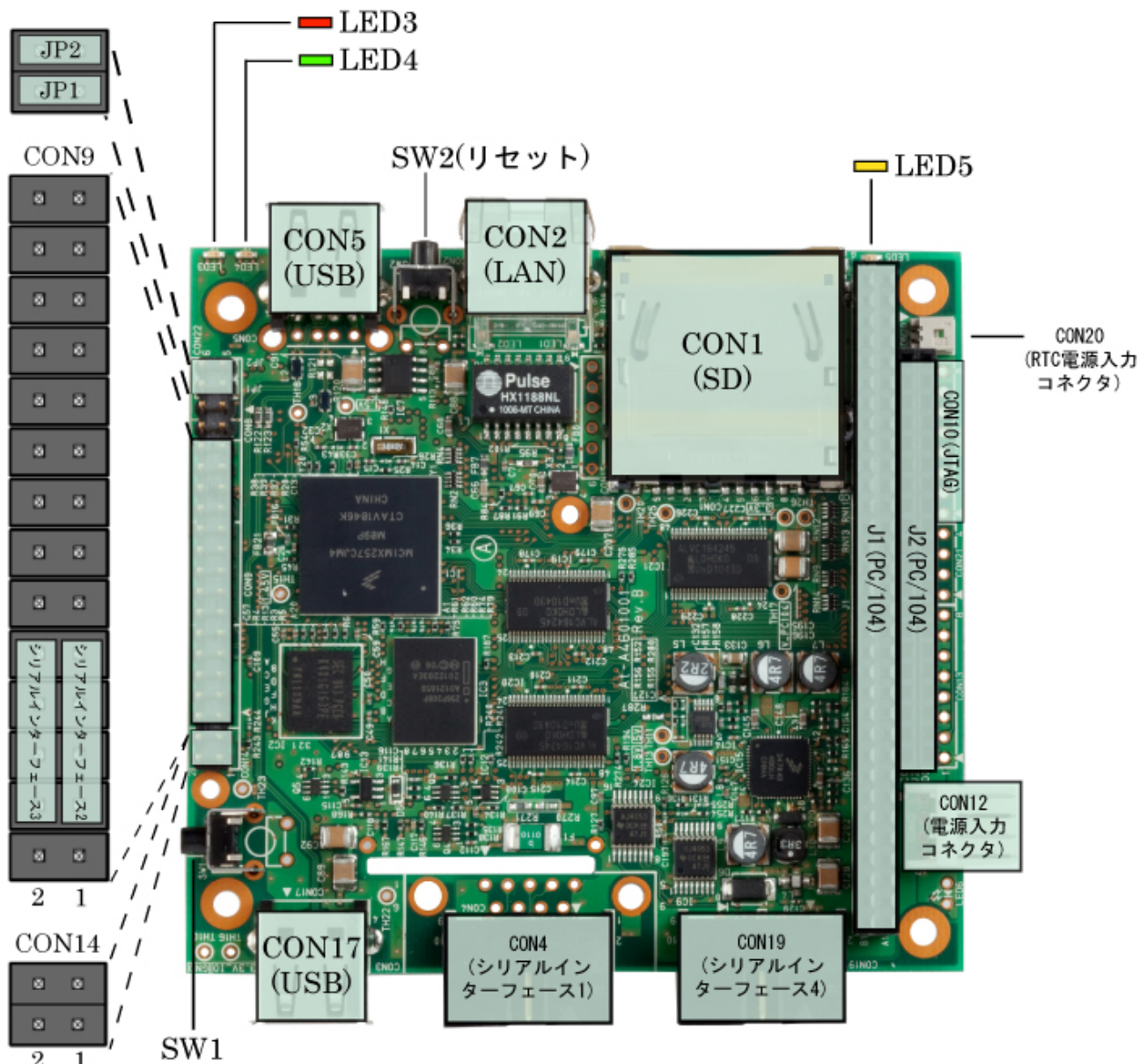


Figure 3.7. Basic Layout of Armadillo-460 Basic Model

Table 3.10. Default States of Armadillo-460 Basic Model Expansion Interfaces

Pin Number	Function	Input/Output	Open Drain	Pull/Keeper ^[a]	Slew Rate	Drive Strength
CON14 1	+3.3V_IO	Power	-	-	-	-
CON14 2	GND	Power	-	-	-	-
CON14 3	I2C2_SCL	Input/Output	Enabled	22kΩ PU	Slow	Std.
CON14 4	I2C2_SDA	Input/Output	Enabled	22kΩ PU	Slow	Std.
CON9 1	GPIO3_17	Input	Disabled	100kΩ PU	Slow	Std.
CON9 2	GPIO3_14	Input	Disabled	100kΩ PU	Slow	Std.
CON9 3	Serial Interface 2 UART3_RXD	Input	Disabled	100kΩ PU	Slow	Std.

Pin Number	Function	Input/Output	Open Drain	Pull/Keeper ^[a]	Slew Rate	Drive Strength
CON9 4	Serial Interface 3 UART5_RXD	Input	Disabled	100kΩ PU	Slow	Std.
CON9 5	Serial Interface 2 UART3_TXD	Output	Disabled	Disabled	Slow	Std.
CON9 6	Serial Interface 3 UART5_TXD	Output	Disabled	Disabled	Slow	Std.
CON9 7	+3.3V_IO	Power	-	-	-	-
CON9 8	+3.3V_IO	Power	-	-	-	-
CON9 9	GND	Power	-	-	-	-
CON9 10	GND	Power	-	-	-	-
CON9 11	GPIO1_17	Input	Disabled	100kΩ PU	Slow	Std.
CON9 12	GPIO1_29	Input	Disabled	100kΩ PU	Slow	Std.
CON9 13	GPIO1_18	Input	Disabled	100kΩ PU	Slow	Std.
CON9 14	GPIO1_30	Input	Disabled	100kΩ PU	Slow	Std.
CON9 15	GPIO1_7	Input	Disabled	100kΩ PU	Slow	Std.
CON9 16	GPIO1_31	Input	Disabled	100kΩ PU	Slow	Std.
CON9 17	GPIO4_21	Input	Disabled	100kΩ PU	Slow	Std.
CON9 18	GPIO1_6	Input	Disabled	100kΩ PU	Slow	Std.
CON9 19	GND	Power	-	-	-	-
CON9 20	+3.3V_IO	Power	-	-	-	-
CON9 21	GPIO1_8	Input	Disabled	100kΩ PU	Slow	Std.
CON9 22	GPIO1_9	Input	Disabled	100kΩ PU	Slow	Std.
CON9 23	GPIO1_10	Input	Disabled	100kΩ PU	Slow	Std.
CON9 24	GPIO1_11	Input	Disabled	100kΩ PU	Slow	Std.
CON9 25	GPIO1_16	Input	Disabled	100kΩ PU	Slow	Std.
CON9 26	GPIO2_22	Input	Disabled	100kΩ PU	Slow	Std.
CON9 27	GPIO2_21	Output Low	Disabled	Disabled	Fast	Std.
CON9 28	GPIO3_15	Output Low	Disabled	Disabled	Fast	Std.
CON11 1	VIN	Power	-	-	-	-
CON11 2	VIN	Power	-	-	-	-
CON11 3	VIN	Power	-	-	-	-
CON11 4	+3.3V_IO	Power	-	-	-	-
CON11 5	+3.3V_IO	Power	-	-	-	-
CON11 6	GND	Power	-	-	-	-
CON11 7	GND	Power	-	-	-	-
CON11 8	LCDC_LSCLK	Output	Disabled	Disabled	Fast	Std.
CON11 9	LCDC_HSYNC	Output	Disabled	Disabled	Slow	Std.
CON11 10	LCDC_VSYNC	Output	Disabled	Disabled	Slow	Std.
CON11 11	LCDC_OE_ACD	Output	Disabled	Disabled	Slow	Std.
CON11 12	PWM1_PWM0	Output	Disabled	100kΩ PU	Slow	Std.
CON11 13	LCDC_LD0	Output	Disabled	Disabled	Slow	Std.
CON11 14	LCDC_LD1	Output	Disabled	Disabled	Slow	Std.
CON11 15	LCDC_LD2	Output	Disabled	Disabled	Slow	Std.
CON11 16	LCDC_LD3	Output	Disabled	Disabled	Slow	Std.
CON11 17	LCDC_LD4	Output	Disabled	Disabled	Slow	Std.
CON11 18	LCDC_LD5	Output	Disabled	Disabled	Slow	Std.
CON11 19	GND	Power	-	-	-	-
CON11 20	LCDC_LD6	Output	Disabled	Disabled ^[b]	Slow	Std.
CON11 21	LCDC_LD7	Output	Disabled	Disabled	Slow	Std.
CON11 22	LCDC_LD8	Output	Disabled	Disabled	Slow	Std.
CON11 23	LCDC_LD9	Output	Disabled	Disabled	Slow	Std.
CON11 24	LCDC_LD10	Output	Disabled	Disabled	Slow	Std.
CON11 25	LCDC_LD11	Output	Disabled	Disabled	Slow	Std.
CON11 26	GND	Power	-	-	-	-

Pin Number	Function	Input/Output	Open Drain	Pull/Keeper ^[a]	Slew Rate	Drive Strength
CON11 27	LCDC_LD12	Output	Disabled	Disabled	Slow	Std.
CON11 28	LCDC_LD13	Output	Disabled	Disabled	Slow	Std.
CON11 29	LCDC_LD14	Output	Disabled	Disabled ^[b]	Slow	Std.
CON11 30	LCDC_LD15	Output	Disabled	Disabled ^[b]	Slow	Std.
CON11 31	LCDC_LD16	Output	Disabled	Disabled	Slow	Std.
CON11 32	LCDC_LD17	Output	Disabled	Disabled	Slow	Std.
CON11 33	GND	Power	-	-	-	-
CON11 34	ADC_XP	Analog Input	-	-	-	-
CON11 35	ADC_XN	Analog Input	-	-	-	-
CON11 36	ADC_YP	Analog Input	-	-	-	-
CON11 37	ADC_YN	Analog Input	-	-	-	-
CON11 38	GND	Power	-	-	-	-
CON11 39	LCD_SW1 (GPIO2_20)	Input	Disabled	47kΩ PU	Slow	Std.
CON11 40	LCD_SW2 (GPIO2_29)	Input	Disabled	47kΩ PU	Slow	Std.
CON11 41	LCD_SW3 (GPIO2_30)	Input	Disabled	47kΩ PU	Slow	Std.
CON11 42	GPIO2_31	Input	Disabled	100kΩ PU	Slow	Std.
CON11 43	GPIO3_0	Input	Disabled	100kΩ PU	Slow	Std.
CON11 44	N.C. ^[c]	-	-	-	-	-
CON11 45	N.C. ^[d]	-	-	-	-	-
CON11 46	N.C. ^[e]	-	-	-	-	-
CON11 47	N.C. ^[f]	-	-	-	-	-
CON11 48	I2C3_SCL	Input/Output	Enabled	22kΩ PU	Slow	Std.
CON11 49	I2C3_SDA	Input/Output	Enabled	22kΩ PU	Slow	Std.
CON11 50	GND	Power	-	-	-	-

^[a]PD=pull-down, PU=pull-up.


^[b]47kΩ PU on the Armadillo-440 board.

^[c]Can be connected to AUD5_TXD/UART4_RXD on i.MX257 by altering the CPLD Ext I/F Control Register.

^[d]Can be connected to AUD5_RXD/UART4_TXD on i.MX257 by altering the CPLD Ext I/F Control Register.

^[e]Can be connected to AUD5_TXC/UART4_RTS on i.MX257 by altering the CPLD Ext I/F Control Register.

^[f]Can be connected to AUD5_TXFS/UART4_CTS on i.MX257 by altering the CPLD Ext I/F Control Register.



Serial Interfaces 2 and 3 have +3.3V I/O levels. They can be used at RS232C levels by connecting the optional^[9] RS232C level conversion adapter.

When using the RS232C level conversion adapter, please connect pin 1 (the yellow or green wire) to CON9 1 for Serial Interface 2 and to CON9 2 for Serial Interface 3.

3.7. Memory Map

The default partitioning of flash memory on the Armadillo-400 Series boards is shown in Table 3.11, “Armadillo-420 Flash Memory Map” and Table 3.12, “Armadillo-440/460 Flash Memory Memory Map”.

Table 3.11. Armadillo-420 Flash Memory Map

Physical Address	Region Name	Size	Description
0xa0000000 0xa001ffff	bootloader	128KB	Bootloader image is stored here

^[9]A RS232C level conversion adapter is available as an option and is also included in the development set.

Physical Address	Region Name	Size	Description
0xa0020000 0xa021ffff	kernel	2MB	Kernel image is stored here
0xa0220000 0xa0fdffff	userland	13.75MB	Userland image is stored here
0xa0fe0000 0xa0ffffff	config	128KB	Configuration data is stored here

Table 3.12. Armadillo-440/460 Flash Memory Memory Map

Physical Address	Region Name	Size	Description
0xa0000000 0xa001ffff	bootloader	128KB	Bootloader image is stored here
0xa0020000 0xa021ffff	kernel	2MB	Kernel image is stored here
0xa0220000 0xa1fdffff	userland	29.75MB	Userland image is stored here
0xa1fe0000 0xa1ffffff	config	128KB	Configuration data is stored here

3.8. Software Make-up

The Armadillo-400 Series operates with the software described below.

3.8.1. Bootloader

The bootloader is the first software program to run after the board is turned on. The Hermit-At Bootloader (hereafter referred to as Hermit-At) is used on the Armadillo-400 Series.

Hermit-At has two operating modes: auto-boot mode and maintenance mode. In auto-boot mode, the kernel image is loaded to RAM from a predetermined place and then booted. In maintenance mode is it possible to carry out operations such as updating flash memory and setting boot options. For more information, please refer to Appendix A, Hermit-At Bootloader.

The bootloader must be stored in the bootloader region of flash memory.

3.8.2. Kernel

Linux 2.6 is used as the default kernel on the Armadillo-400 Series.

The kernel image is stored in the kernel region of flash memory by default. It is also possible to use a kernel image from storage (microSD/SD) or from a TFTP server by altering Hermit-At's boot options.

3.8.3. Userland

The standard userland root filesystem used on the Armadillo-400 Series is an initrd^[10] image created from a source code based distribution named Atmark-Dist.

^[10]Initial RAM disk. On standard Linux systems, an initrd is used as a temporary "mini" root filesystem before the root file system stored in external (HDD etc) storage is mounted. On the Armadillo-400 Series, the initrd is used as the final root file system.

In addition to the standard userland, a Debian GNU/Linux based userland option is also available.

By default, the initrd image is stored in the userland region of flash memory and loaded as a RAM disk by Hermit-At. It is possible to use an image from a TFTP server by altering Hermit-At's boot options.

Aside from a RAM disk, it is also possible to use a root filesystem from external storage (microSD/SD/USB) or a NFS server^[11] by setting the appropriate kernel parameters.

The use of a kernel or userland image not stored in flash memory is explained in Chapter 8, Kernel and Userland Placement.

3.8.4. Downloader

This is an application that runs on the work PC and is used to write to the flash memory on the Armadillo.

For Linux PCs, the Hermit-At downloader and Shoehorn-At are available. The Hermit-At downloader works in cooperation with the target Armadillo to rewrite the on-board flash memory. Shoehorn-At is used to restore the bootloader.

The downloader available for Windows PCs is called Hermit-At Win32. Hermit-At Win32 can be used to both rewrite the on-board flash memory and restore the bootloader on the Armadillo.

3.9. Boot Modes

JP1 is used to select between on-board flash memory boot mode and UART boot mode on the Armadillo-400 Series.

In on-board flash memory boot mode, the bootloader stored in the bootloader region of flash memory is executed at power on.

With the default bootloader (Hermit-At), JP2 is used to select between auto boot mode, where the kernel is automatically booted, and maintenance mode, where it is possible to carry out various configuration.


However, even when auto boot mode is selected with JP2, the auto boot cancel function in Hermit-At will cause it to enter maintenance mode if SW1 is depressed at boot time.

The UART boot mode is used for system restore when, for example, the flash memory bootloader has been damaged. For more information, please refer to Section 6.6, "Restoring Bootloader to Factory State".


The jumper settings for each boot mode on the Armadillo-400 Series are shown in Table 3.13, "Jumper Settings".

Table 3.13. Jumper Settings

JP1	JP2	Boot Modes
Open	Open	On-board flash memory boot / auto boot mode
Open	Shorted	On-board flash memory boot / maintenance mode
Shorted	-	UART boot mode



Open and Shorted Jumpers



"Open" is when a jumper socket is not connected to the jumper pins.

^[11]When NFS support is enabled in the kernel.



"Shorted" is when a jumper socket is connected to the jumper pins.

Chapter 4. Before Getting Started

4.1. Preparation

The following equipment is required in order to begin development of embedded systems based on the Armadillo-400 Series.

Work PC	A PC that runs either Debian GNU/Linux or Windows and has at least one serial port.
Serial Cross Cable	A D-Sub 9 pin (female-to-female) cable to connect the Armadillo and the work PC.
Serial Console Software	Please install a serial console program on the work PC such as minicom on Linux or Tera Term Pro on Windows

The following equipment is not an absolute requirement, but does allow for improved development efficiency.

LAN Cable	This is required in order to communicate with the Armadillo via LAN. Please connect the Armadillo and work PC via a switching hub ^[1] .
-----------	--

4.2. Connections

Please connect the work PC and peripherals to the Armadillo by referring to the examples in Figure 4.1, “Armadillo-420 Basic Model Connections”, Figure 4.2, “Armadillo-420 WLAN Model (AWL12 Compatible) Connections”, Figure 4.3, “Armadillo-420 WLAN Model (AWL13 Compatible) Connections”, Figure 4.4, “Armadillo-440 LCD Model Connections” or Figure 4.5, “Armadillo-460 Basic Model Connections”.

^[1]As Auto MDIX is supported on the Armadillo-400 Series, it is also possible to connect the Armadillo and work PC directly with a LAN cable.

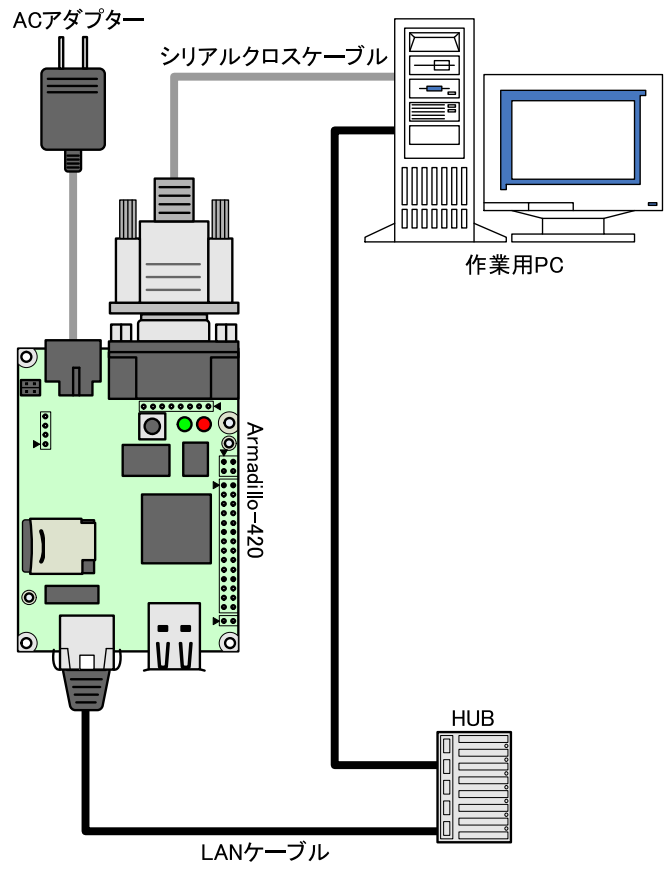


Figure 4.1. Armadillo-420 Basic Model Connections

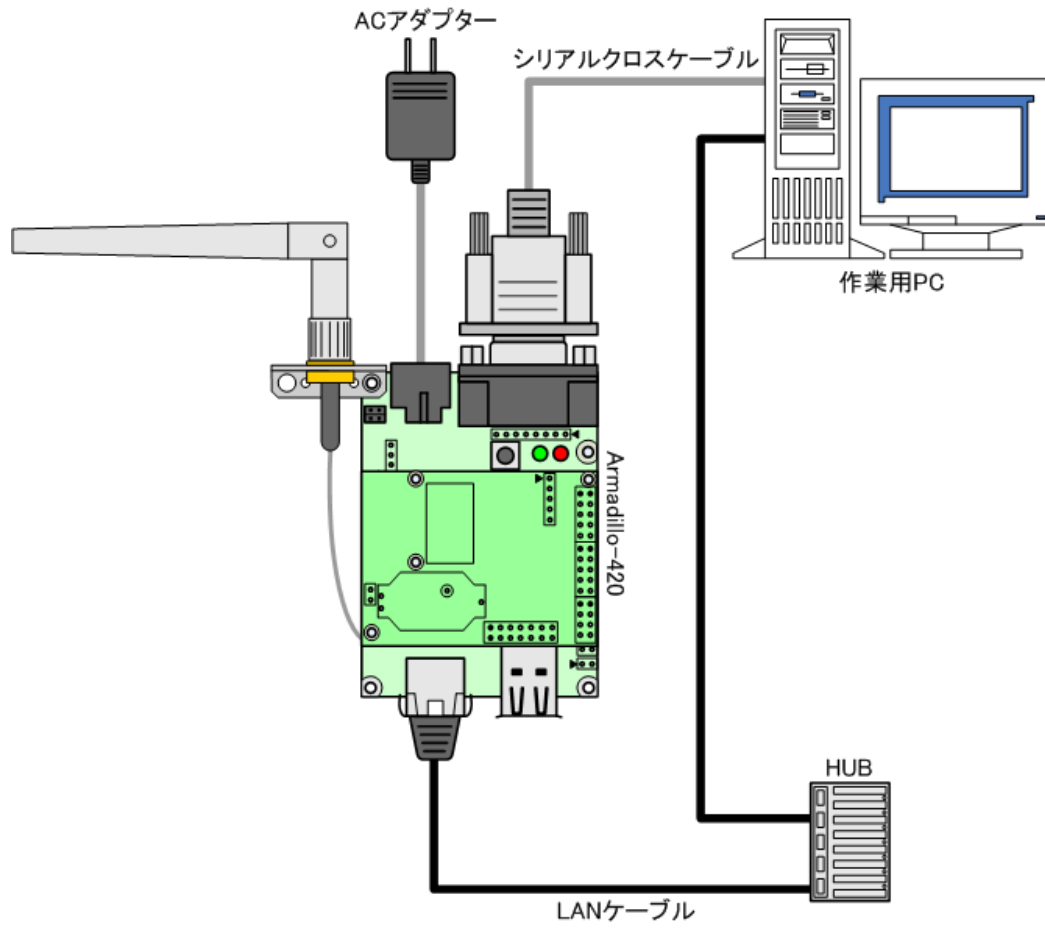


Figure 4.2. Armadillo-420 WLAN Model (AWL12 Compatible) Connections

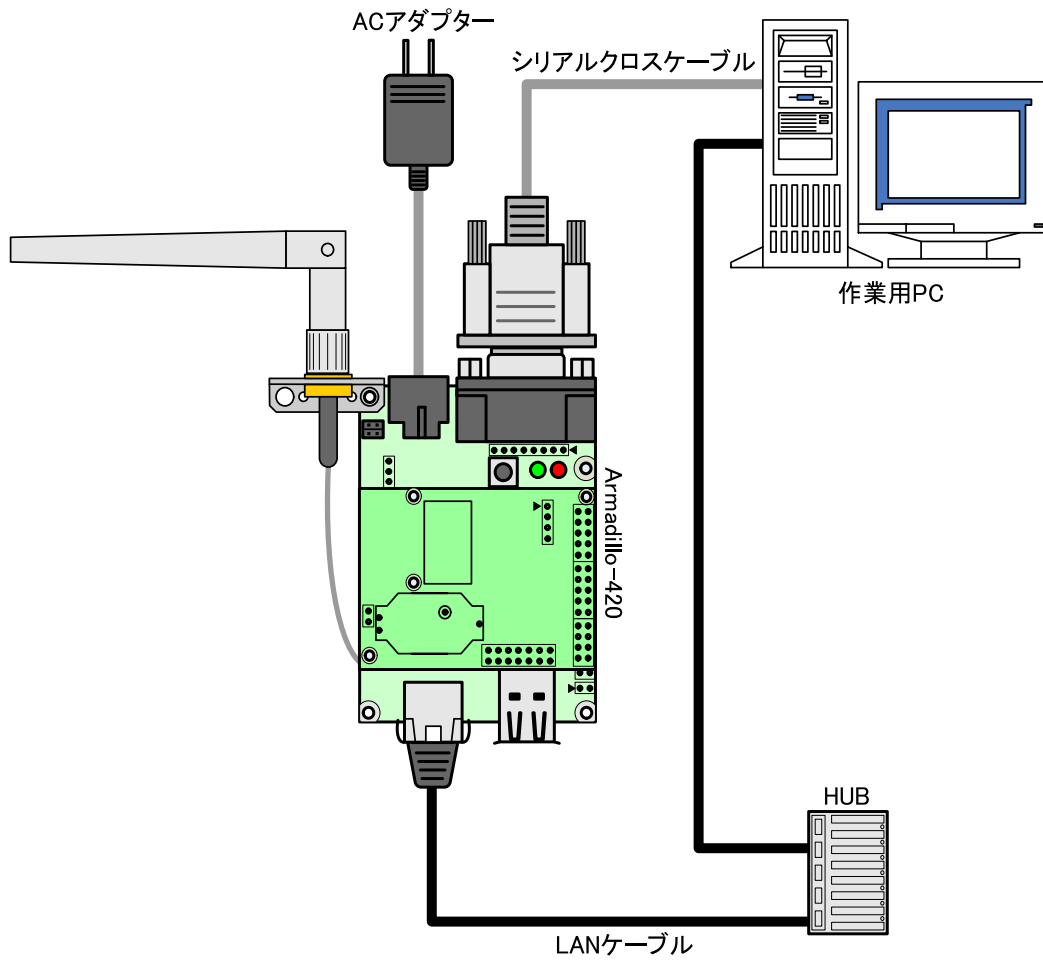


Figure 4.3. Armadillo-420 WLAN Model (AWL13 Compatible) Connections

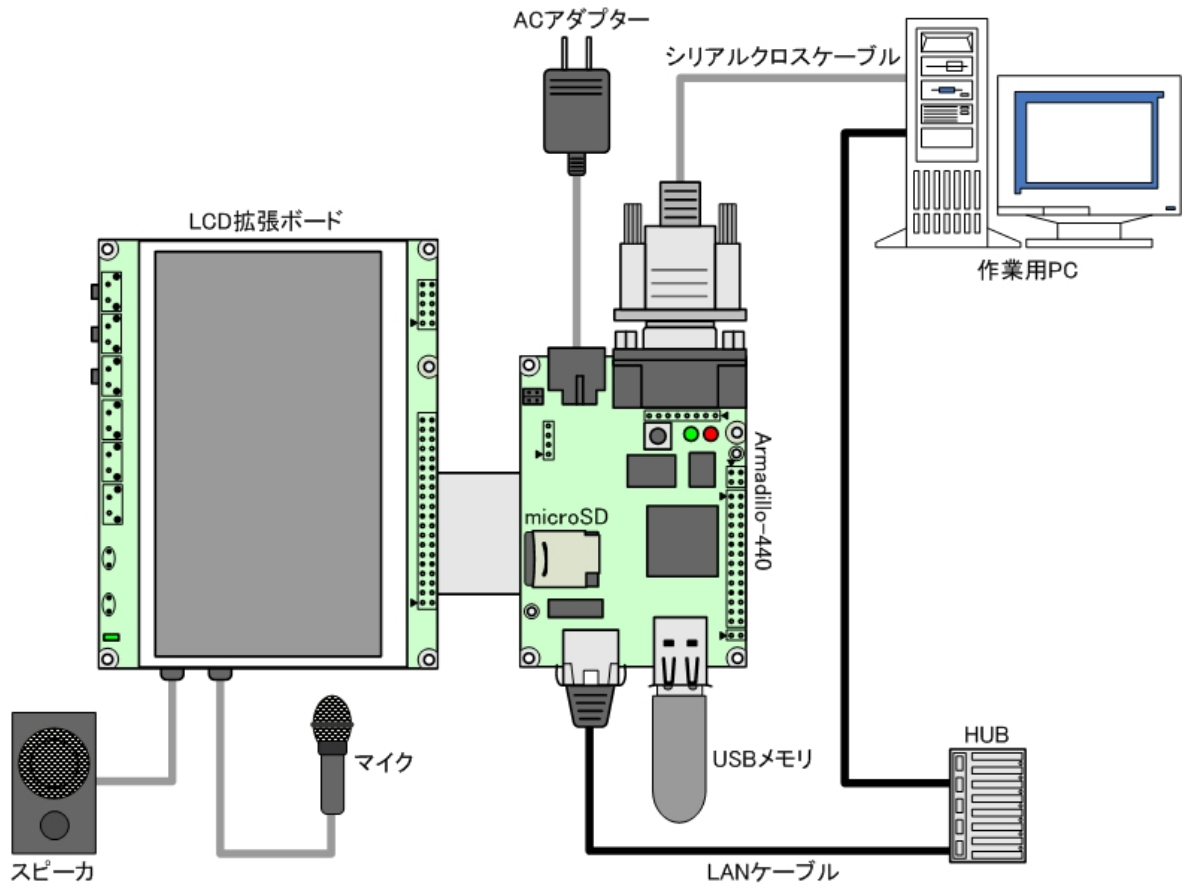


Figure 4.4. Armadillo-440 LCD Model Connections

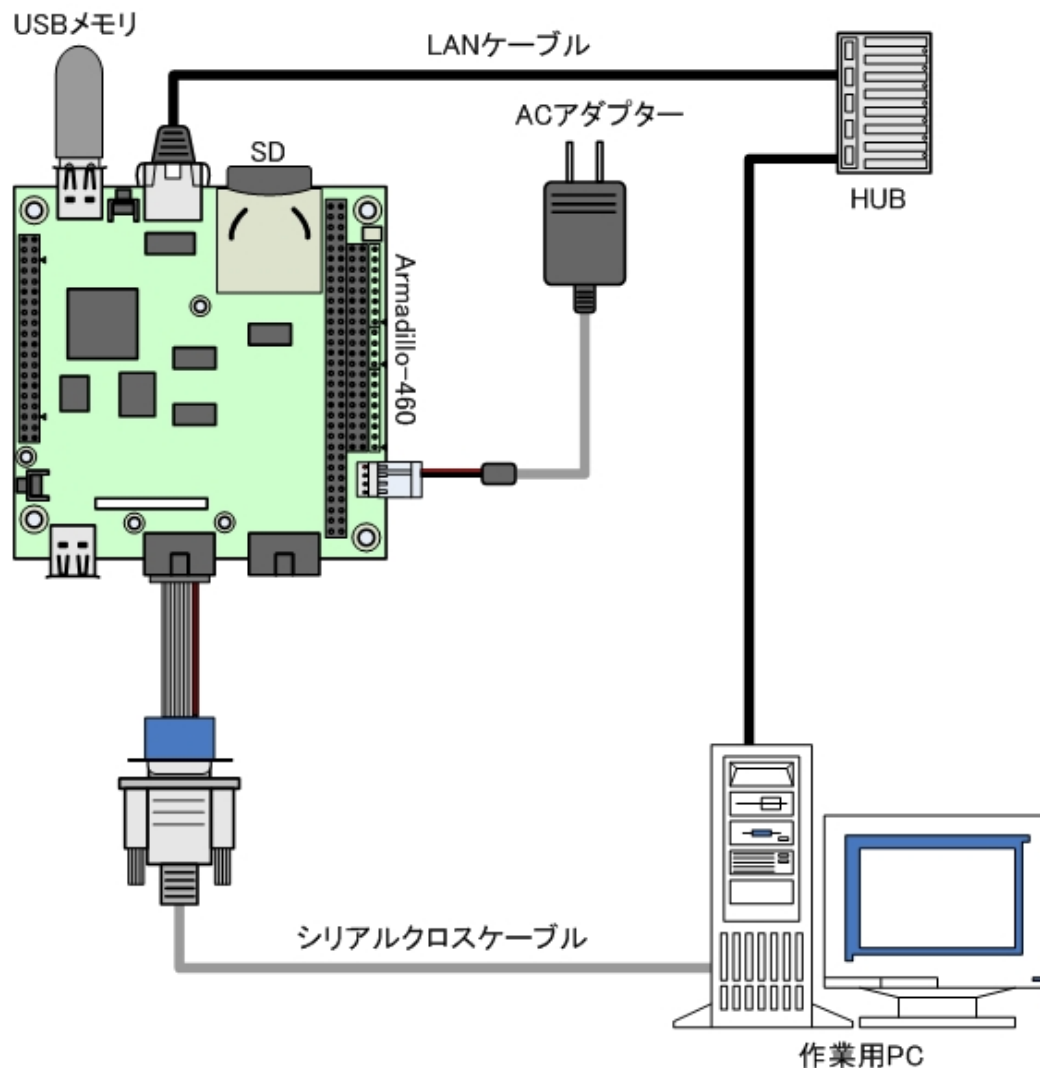


Figure 4.5. Armadillo-460 Basic Model Connections

4.3. Serial Console Software Configuration

When connecting the work PC to a serial console on the Armadillo, please configure the serial communication software with the settings shown in Table 4.1, “Serial Communication Configuration”. Also, please keep the width of the serial communication software to more than 80 characters. The display may become disordered when entering commands if the width is less than 80 characters.

Table 4.1. Serial Communication Configuration

Item	Configuration
Transmission Rate	115,200 bps
Data Length	8 bit
Stop Bit	1 bit
Parity	None
Flow Control	None

Chapter 5. Development Environment Set-up

This chapter explains how to set up a software development environment for Armadillo on a work PC.

A Debian based Linux environment^[1] (Debian/GNU Linux 5.0 codename Lenny is standard) is required to undertake software development for the Armadillo-400 Series.

If the work PC is Windows, a virtual Linux environment must be set up within Windows.

"VMware" is the recommended way to set up a Linux environment on Windows. An operating system image called ATDE (Atmark Techno Development Environment)^[2] is available with the required development software pre-installed.

Please refer to the "ATDE Install Guide" for information on setting up a Linux environment on Windows.

As the basic development environment is pre-installed, there is no need to carry out the operations listed in Section 5.1, "Installing Cross Development Environment Packages" and Section 5.2, "Installing Packages Required for Atmark-Dist Builds" when using ATDE.

5.1. Installing Cross Development Environment Packages

Debian (deb) packages are used for application and library management on Debian based Linux systems.

In order to carry out cross development, toolchain packages for cross development and library packages for the target architecture which have been converted for cross development use must be installed on the work PC.

On Debian based Linux systems there are two ARM architectures: arm and armel. The difference between these two lies in the ABI (Application Binary Interface), with the arm architecture using OABI, and the armel architecture using EABI.

As EABI is the standard ABI on the Armadillo-400 Series, Debian armel packages must be installed.

The cross development environment packages are stored on the included DVD under the `cross-dev/deb/` directory. For development with the Armadillo-400 Series, please install the ARM EABI cross development packages in the `armel` directory.

The install must be carried out as a root user. The deb packages are installed by executing the command shown in Figure 5.1, "Install Command".

```
[PC ~]$ sudo dpkg --install *.deb
```

Figure 5.1. Install Command



sudo is used to execute the specified command as a different user. The above command line executes the **dpkg** command as the root user.

^[1]While it is possible to use Linux distributions other than Debian, not all operations listed in this document will work in the same way. The reader will be required to modify the listed operations to suit their Linux environment when using other distributions.

^[2]ATDE v3.0 or later is the recommended development environment for the Armadillo-400 Series.

A password is requested after executing the **sudo** command. This is the password of the user executing the command, and not the root password.



If a cross development environment for the same target architecture has been previously installed in the Linux environment, please make sure to uninstall this existing environment before installing the new cross development environment.

5.2. Installing Packages Required for Atmark-Dist Builds

Atmark-Dist is the default distribution for Armadillo. The packages listed in Table 5.1, “List of Packages Required for Atmark-Dist Builds” must be installed on the work PC in order to build Atmark-Dist.

Table 5.1. List of Packages Required for Atmark-Dist Builds

Package Name	Version
genext2fs	1.4.1-2.1 or later
file	4.26-1 or later
sed	4.1.5-6 or later
perl	5.10.0-19lenny2 or later
bison	1:2.3.dfsg-5 or later
flex	2.5.35-6 or later
libncurses5-dev	5.7+20081213-1 or later

The currently installed version can be displayed by specifying the package name when executing the command shown in Figure 5.2, “Installed Version Display Command”.

`--list` is an option to **dpkg** used to display package information. The package name pattern for the package to be displayed is specified in place of `package-name-pattern`.

```
[PC ~]$ dpkg --list [package-name-pattern]
```

Figure 5.2. Installed Version Display Command

5.3. Installing Cross Development Library Packages

When building applications or libraries not included in Atmark-Dist, library packages that are neither available in the included DVD or from the download site may be required. The following is an introduction on how to create and install the required cross development library packages.

First, the library package on which the cross development package will be based on must be acquired. The package architecture must match the target and the Debian distribution version should match that of the development environment. For the Armadillo-400 Series, the architecture is `armel` and the Debian distribution version is `lenny` (the oldstable version as at March, 2011).

For example, for the **libjpeg62** library the package name will be `libjpeg62_[version]_armel.deb`.

Debian packages can be searched for and downloaded from the Debian Packages site^[3].

^[3]<http://www.debian.org/distrib/packages>

Once the library package has been obtained, it is then converted for cross development use with the **dpkg-cross** command.

```
[PC ~]$ dpkg-cross --build --arch armel libjpeg62_[version]_armel.deb
[PC ~]$ ls
libjpeg62-armel-cross_[version]_all.deb libjpeg62_[version]_armel.deb
```

Figure 5.3. Cross Development Library Package Creation

The created cross development library package is then installed.

```
[PC ~]$ sudo dpkg -i libjpeg62-armel-cross_[version]_all.deb
```

Figure 5.4. Installing Cross Development Library Packages



If the **dpkg-cross** command is used in environments other than Debian Lenny, there may be times when an installable package cannot be created.

By using **apt-cross**, all of the above steps can be performed with just one command.

```
[PC ~]$ apt-cross --arch armel --suite lenny --install libjpeg62
```

Figure 5.5. apt-cross Command

The target architecture is specified after the **--arch** option, and the Debian distribution version after **--suite**. When **--install** is specified, **apt-cross** automatically installs the converted package. The package name is specified as the last parameter.


Chapter 6. Rewriting Flash Memory

This chapter explains the steps required to rewrite the on-board flash memory on the Armadillo.

There are two main ways to rewrite the flash memory.

1. Rewriting flash memory by using a downloader program on the work PC to send an image to the target Armadillo.
2. Rewriting flash memory by having the target Armadillo download an image file from a remote server.

The first method is explained in Section 6.3, “Rewriting Flash Memory with a Downloader”. The second method is then explained in Section 6.4, “Rewriting Flash Memory with tftpd” and Section 6.5, “Rewriting Flash Memory with netflash”.



If for some reason flash rewriting fails, the software on the Armadillo may no longer be able to boot properly. Be sure to pay attention to the following points when rewriting flash.

- Do not cut power to the Armadillo while rewriting
- Do not disconnect the serial or LAN cable connecting the Armadillo and work PC while rewriting

If the Armadillo can no longer be booted after a rewrite of the bootloader has failed, please follow the steps listed in Section 6.6, “Restoring Bootloader to Factory State” to restore the bootloader.


6.1. Flash Memory Writing Regions

The flash memory address to be written to can be specified by region name. The image files to be specified for each region are shown in Table 6.1, “Region Names and Corresponding Image Files”.

Table 6.1. Region Names and Corresponding Image Files

Product	Region Name	File Name
Armadillo-420 Basic Model	bootloader	loader-armadillo4x0-[version].bin
	kernel	linux-a400-[version].bin.gz
	userland	romfs-a420-[version].img.gz
Armadillo-420 WLAN Model (AWL12 Compatible)	bootloader	loader-armadillo4x0-[version].bin
	kernel	linux-a400-wlan-[version].bin.gz
	userland	romfs-a420-wlan-awl12-[version].img.gz
Armadillo-420 WLAN Model (AWL13 Compatible)	bootloader	loader-armadillo4x0-[version].bin
	kernel	linux-a400-wlan-[version].bin.gz
	userland	romfs-a420-wlan-awl13-[version].img.gz
Armadillo-440 LCD Model	bootloader	loader-armadillo4x0-[version].bin
	kernel	linux-a400-[version].bin.gz
	userland	romfs-a440-[version].img.gz

Product	Region Name	File Name
Armadillo-460 Basic Model	bootloader	loader-armadillo4x0-[<i>version</i>].bin
	kernel	linux-a460-[<i>version</i>].bin.gz
	userland	romfs-a460-[<i>version</i>].img.gz



All models use the same image file for the bootloader. The Armadillo-420 Basic Model and the Armadillo-440 LCD Model both use the same kernel image file.


6.2. Installing Downloader

The downloader program is installed on the work PC.

There are multiple types of downloader, as listed in Table 6.2, “Downloader List”.

Table 6.2. Downloader List

Downloader	Operating System	Description
Hermit-At Downloader	Linux	This is a CUI application for Linux.
Shoehorn-At	Linux	This is a CUI application for Linux.
Hermit-At Win32	Windows	This is a GUI application for Windows.



The downloader package is pre-installed in ATDE (Atmark Techno Development Environment), so there is no need to install it separately.

6.2.1. For Linux

Install the package files contained in the `downloader/` directory on the included DVD.

```
[PC ~]$ sudo dpkg --install hermit-at_[version].i386.deb
[PC ~]$ sudo dpkg --install shoehorn-at_[version].i386.deb
```

Figure 6.1. Installing Downloader (Linux)

6.2.2. For Windows

Extract the contents of the `hermit-at-win_[version].zip` archive stored in the `downloader/` directory on the included DVD to an appropriate folder.

6.3. Rewriting Flash Memory with a Downloader

The following explains how to rewrite flash memory with the Hermit-At Downloader and Hermit-At Win32.

The Hermit-At Downloader and Hermit-At Win32 work in cooperation with the bootloader on the Armadillo to rewrite the flash memory from the work PC.

6.3.1. Preparation

Boot the Armadillo in maintenance mode by appropriately setting the jumpers as shown in Table 3.13, “Jumper Settings” and then powering the board on.

Ensure that the serial interface on the work PC that is connected to the Armadillo is not being used by another application. If another application is using the serial interface, please either close the connection or close the application in order to free the interface.

6.3.2. For Linux

On Linux, execute the **hermit-at** command as shown in Figure 6.2, “Download Command”.

The download command is a sub-command of **hermit-at**. It is used to write the file specified with the `--input-file` option to the target board. The `--region` option is used to specify the region to be written to. Therefore in the following example `linux.bin.gz` will be written to the kernel region.

```
[PC ~]$ hermit download --input-file linux.bin.gz --region kernel
```

Figure 6.2. Download Command

If using a serial interface other than `ttyS0`, the port must be specified with the `--port` option as shown in Figure 6.3, “Download Command (with Port Option)”^[1].

```
[PC ~]$ hermit download --input-file linux.bin.gz --region kernel --port ttyS1
```

Figure 6.3. Download Command (with Port Option)

The bootloader region has a simple protection mechanism to prevent mistaken rewrites. To rewrite the bootloader region, override the protection mechanism by using the `--force-locked` option as shown in Figure 6.4, “Download Command (Unprotected)”^[1].

```
[PC ~]$ hermit download --input-file loader-armadillo4x0-[version].bin
--region bootloader --force-locked
```

Figure 6.4. Download Command (Unprotected)



If an incorrect image is written to the bootloader region, it will no longer be possible to boot from on-board flash memory. If this does occur, restore the bootloader by referring to Section 6.6, “Restoring Bootloader to Factory State”.

6.3.3. For Windows

For Windows, execute `hermit.exe`. The window shown in Figure 6.5, “Hermit-At Win32: Download Window” will appear.

^[1]The command is only shown as multiple lines due to formatting constraints and should be entered as one line.

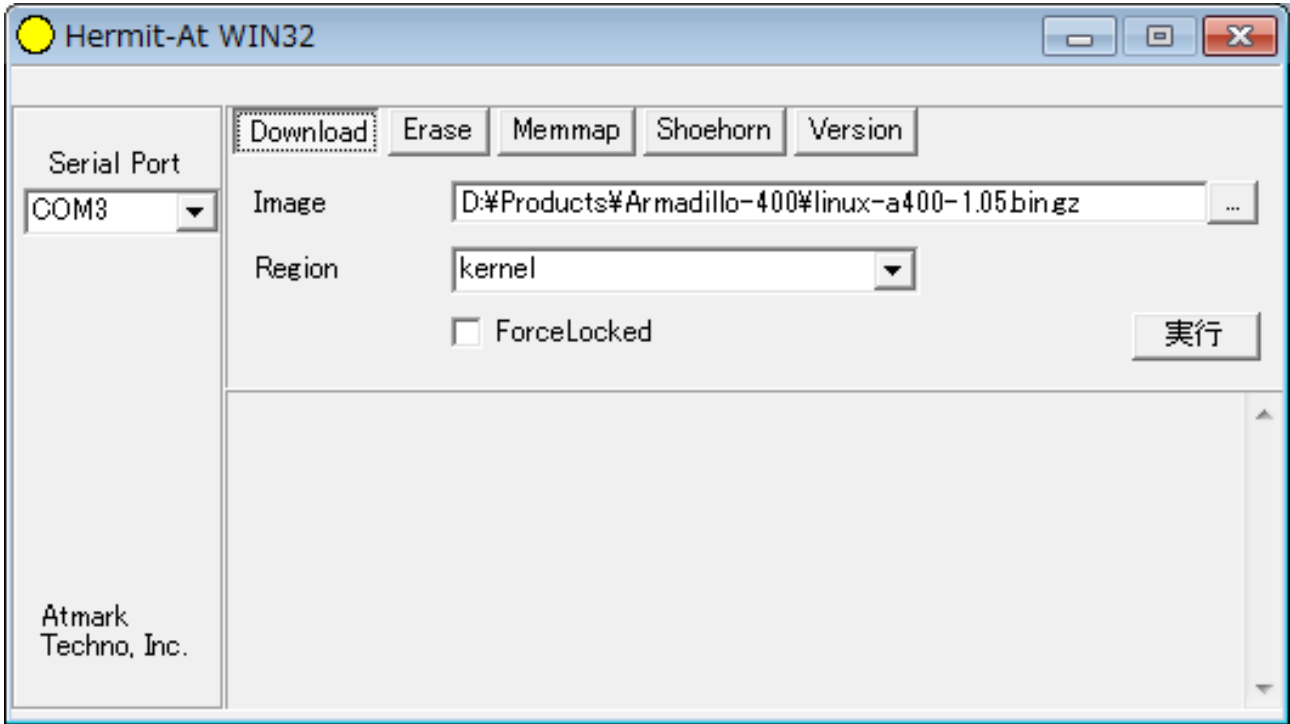


Figure 6.5. Hermit-At Win32: Download Window

Under "Serial Port", select the serial interface connected to the Armadillo. If a drop-down list is not displayed, please enter the port name directly.

For "Image", specify the image to be written and for "Region", specify the region to be written to. If specifying the all or bootloader regions, "Force Locked" must be selected.

After all configuration is complete, click the Execute button to start the writing. The download progress is displayed in a dialog box as shown in Figure 6.6, "Hermit-At Win32: Download Dialog" while the image is being written to flash.

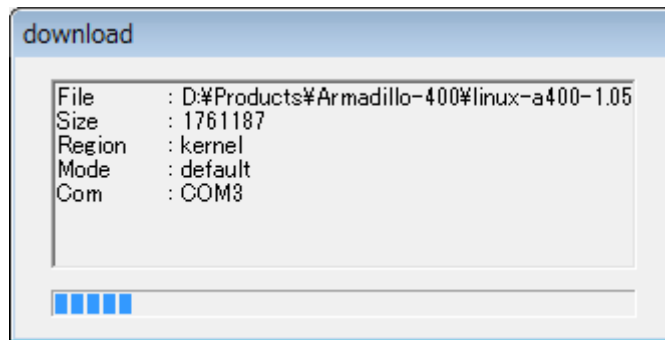


Figure 6.6. Hermit-At Win32: Download Dialog


The dialog box will close once the download has finished.

6.4. Rewriting Flash Memory with tftpd

The following explains how to rewrite flash memory by having the target Armadillo download an image from a remote server.

Using the `tftpd` function of the Hermit-At bootloader is a faster way to rewrite flash memory than using a downloader program.

The `tftpd` function gives the Armadillo the ability to download an image made available by a TFTP server on the same network and then write the image to its own flash memory.



A TFTP server (`atftpd`) is started by default on ATDE v3.0 and later. Files placed in the `/var/lib/tftpboot/` directory will be accessible via TFTP.

In order to use the `tftpd` function, first configure the jumpers on the target Armadillo for maintenance mode and boot the board.

Use the serial console program on the work PC to enter the `tftpd` command. In the example shown in Figure 6.7, “`tftpd` Command Example”, the Armadillo's IP address is set to 192.168.10.10 and `linux.bin.gz` from a TFTP server with the IP address of 192.168.10.1 is written to the kernel region.

```
hermit> tftpd 192.168.10.10 192.168.10.1 --kernel=linux.bin.gz
```

Figure 6.7. tftpd Command Example

The bootloader, kernel and userland regions can be specified for rewriting. The region names and their corresponding options are shown in Table 6.3, “Region Names and Corresponding Options”.


Table 6.3. Region Names and Corresponding Options

Region	Option
Bootloader	--bootloader
Kernel	--kernel
Userland	--userland

6.5. Rewriting Flash Memory with netflash

It is possible to rewrite flash memory with the `netflash` application once the Armadillo has booted into Linux.

The `netflash` command gives the Armadillo the ability to download an image made available by a HTTP or FTP server on the same network and then write the image to its own flash memory.



A HTTP server (`lighttpd`) is started by default on ATDE v3.0 and later. Files placed in the `/var/www/` directory will be accessible via HTTP.

In order to use the `netflash` command, first login to the Armadillo and then execute the command as shown in Figure 6.8, “`netflash` Command Example”.

```
[armadillo ~]# netflash -k -n -u -r /dev/flash/kernel [URL]
```

Figure 6.8. netflash Command Example

The "-r [device-file-name]" option is used to specify the region to be written to. Please refer to Table 6.4, "Region Names and Corresponding Device Files" for the device file names. Details on other options can be found by executing netflash -h.

Table 6.4. Region Names and Corresponding Device Files

Region	Device File
Kernel	/dev/flash/kernel
Userland	/dev/flash/userland



As the device file of the bootloader region is read-only by default, it cannot be rewritten with **netflash**.

6.6. Restoring Bootloader to Factory State

If for some reason the contents of the bootloader region have been damaged and the bootloader no longer functions properly, the bootloader can be restored to factory state by using the UART boot mode.

6.6.1. Preparation

Configure the jumpers on the Armadillo for UART boot mode by referring to Table 3.13, "Jumper Settings", but do not boot the Armadillo at this stage.

Ensure that the serial interface on the work PC that is connected to the Armadillo is not being used by another application. If another application is using the serial interface, please either close the connection or close the application in order to free the interface.

6.6.2. For Linux

Execute the command^[2] in Figure 6.9, "shoehorn Command Example" and then power on the Armadillo.

```
[PC ~]$shoehorn --boot --target armadillo4x0
--initrd /dev/null
--kernel /usr/lib/hermit/loader-armadillo4x0-boot-[version].bin
--loader /usr/lib/shoehorn/shoehorn-armadillo4x0.bin
--initfile /usr/lib/shoehorn/shoehorn-armadillo4x0.init
--postfile /usr/lib/shoehorn/shoehorn-armadillo4x0.post
```

Figure 6.9. shoehorn Command Example

After executing the command the log shown in Figure 6.10, "shoehorn Command Log" will be displayed.

^[2]The command is only shown as multiple lines due to formatting constraints and should be entered as one line.

```

/usr/lib/shoehorn/shoehorn-armadillo4x0.bin: 1272 bytes (2048 bytes buffer)
/usr/lib/hermit/loader-armadillo4x0-boot-v2.0.0.bin: 45896 bytes (45896 bytes
buffer)
/dev/null: 0 bytes (0 bytes buffer)
Waiting for target - press Wakeup now.
Initializing target...
Writing SRAM loader...
Pinging loader
Initialising hardware:
- flushing cache/TLB
- Switching to 115200 baud
- Initializing for Mobile-DDR
Pinging loader
Detecting DRAM
- 32 bits wide
- start: 0x80000000 size: 0x04000000 last: 0x83ffffff
Total DRAM: 65536kB
Loading /usr/lib/hermit/loader-armadillo4x0-boot-v2.0.0.bin:
- start: 0x83000000 size: 0x0000b348 last: 0x8300b347
initrd_start is c0400000
Moving initrd_start to c0400000
Loading /dev/null:
- start: 0xc0400000 size: 0x00000000
Writing parameter area
- nr_pages (all banks): 4096
- rootdev: (RAMDISK_MAJOR, 0)
- pages_in_bank[0]: 2048
- pages_in_bank[1]: 2048
- initrd_start: 0xc0400000
- initrd_size: 0x0
- ramdisk_size: 0x0
- start: 0x80020000 size: 0x00000900 last: 0x800208ff
Pinging loader
Starting kernel at 0x83000000

```

Figure 6.10. shoehorn Command Log

A successful execution of the **shoehorn** command will result in the UART boot mode version of the Hermit At boot-loader (loader-armadillo4x0-boot-[*version*].bin) running on the Armadillo. The following steps must be done without changing jumper settings or powering off the board.

Write the bootloader to the target as shown in Figure 6.11, “Bootloader Write Command Example”^[3].

```

[PC ~]$ hermit download --input-file loader-armadillo4x0-[version].bin
--region bootloader --force-locked

```

Figure 6.11. Bootloader Write Command Example

6.6.3. For Windows

Execute `hermit.exe` and click on the Shoehorn button. The window shown in Figure 6.12, “Hermit-At Win32: Shoehorn Window” will appear.

^[3]The command is only shown as multiple lines due to formatting constraints and should be entered as one line.

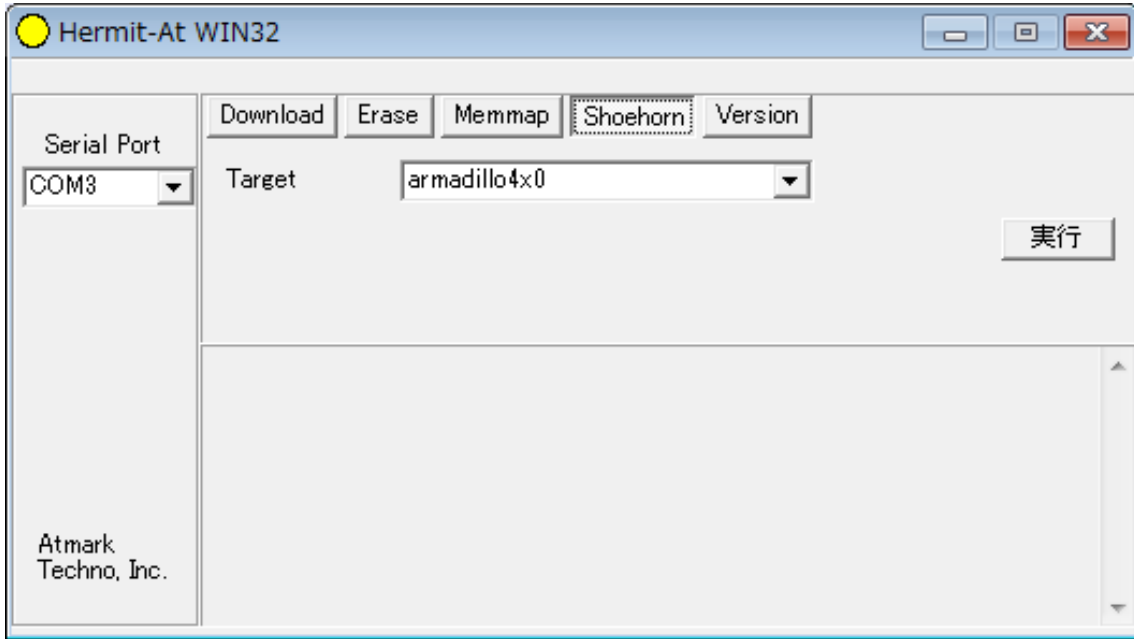


Figure 6.12. Hermit-At Win32: Shoehorn Window

Select armadillo4x0 for the target and click the execute button.

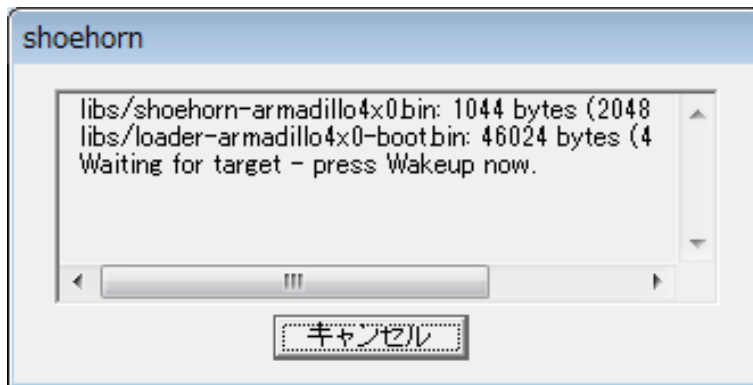



Figure 6.13. Hermit-At Win32: shoehorn Dialog

After the dialog box appears, power on the Armadillo. The dialog box will then close automatically once the Armadillo is ready for downloading. The following steps must be done without changing jumper settings or powering off the board.

Erase the bootloader region once before performing the download. Figure 6.14, “Hermit-At Win32: Erase Window” will be displayed when the "Erase" button is clicked.



Hermit-At Win32 v1.3.0 or later is required to perform the erase. These steps were not supported on Hermit-At Win32 v1.2.0 and previous versions. It is possible to perform the download without erasing first, but parameters saved to flash memory like those saved with the setenv sub-command will not be erased.

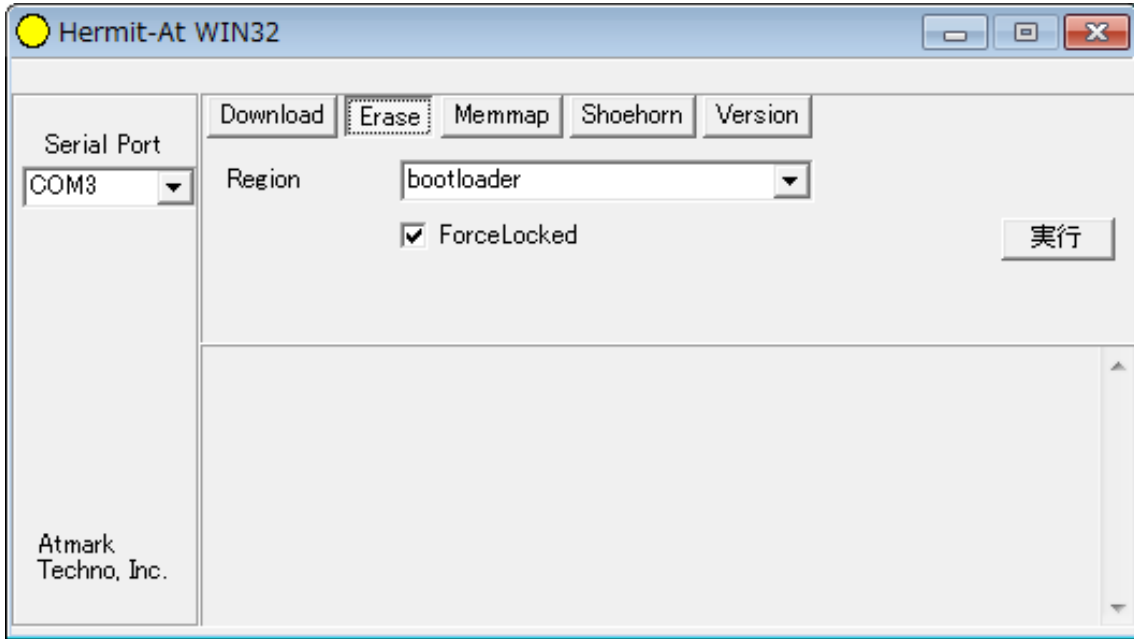


Figure 6.14. Hermit-At Win32: Erase Window

Select the "bootloader" region for "Region", enable "Force Locked" and then click the "Execute" button. While the bootloader region is being erased Figure 6.15, "Hermit-At Win32 : Erase Dialog" will be displayed showing the erase settings and progress.

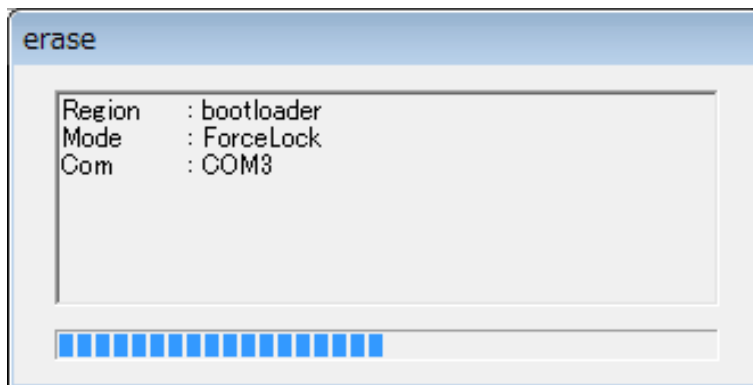


Figure 6.15. Hermit-At Win32 : Erase Dialog

The dialog box will close once the erasing of the bootloader region has completed. Following this, perform the download. Figure 6.16, "Hermit-At Win32: Download Window (After Erase)" will be displayed when the "Download" button is clicked.

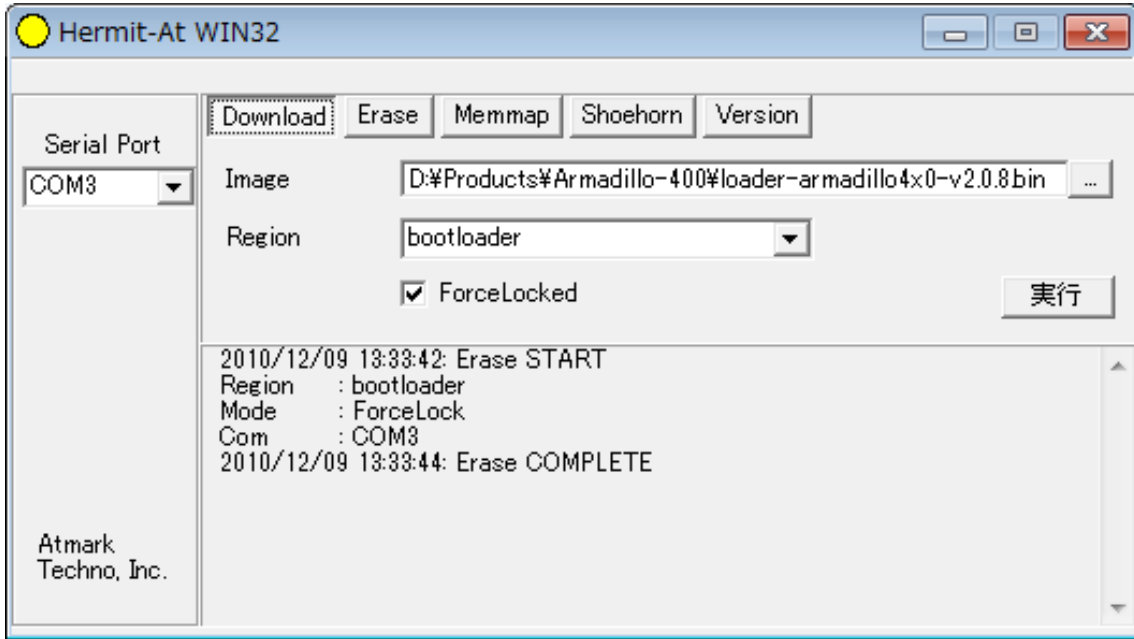


Figure 6.16. Hermit-At Win32: Download Window (After Erase)

Select the bootloader image file for "Image", select "bootloader" for "Region", enable "Force Locked" and then click the "Execute" button. While downloading Figure 6.17, "Hermit-At Win32: Download Dialog (bootloader)" will be displayed showing the download settings and progress.

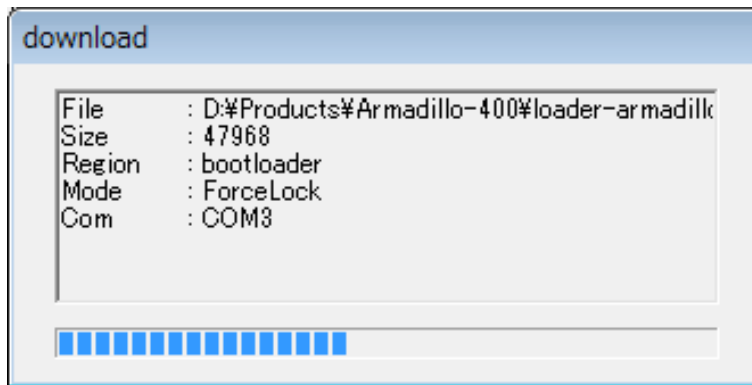


Figure 6.17. Hermit-At Win32: Download Dialog (bootloader)

The dialog box will close once the download has finished.

6.7. Restoring Bootloader Parameters to Factory State

Bootloader parameters are also saved along with the bootloader image in the bootloader flash memory region. This is done so that configuration can be maintained after the Armadillo is reset. The parameters and their initial configuration are shown in Table 6.5, "Bootloader Parameters".

Table 6.5. Bootloader Parameters

Parameter	Initial Configuration	Description
Linux Kernel Parameters	None	Parameters supplied to the Linux kernel at boot time
Boot Device	Flash Memory	Specify the device storing the Linux kernel

In order to restore the bootloader parameters to their factory state, first configure the jumpers on the target Armadillo for maintenance mode and boot the board.

Enter the following commands using the serial communication software on the work PC. To return the Linux kernel parameters to their initial state, execute the command shown in Figure 6.18, “Returning Kernel Parameters to Default State”.^[4]

```
hermit> clearenv
```

Figure 6.18. Returning Kernel Parameters to Default State

Enter the command shown in Figure 6.19, “Returning Boot Device to Default Configuration” to return to using flash memory as the boot device.^[4]

```
hermit> setbootdevice flash
```

Figure 6.19. Returning Boot Device to Default Configuration

^[4]There is no need to perform the following steps if those described in Section 6.6, “Restoring Bootloader to Factory State” are performed first as the parameters are also initialized then. However, with Hermit-AT Win32 v1.2.0 or earlier versions the parameters are not automatically initialized and so the following steps will need to be performed.

Chapter 7. Building

This chapter explains how to build images from source code with the same content as the factory images, and also how to customize these images.

If you have not yet set up a development environment on a work PC, please do so now by referring to Chapter 5, Development Environment Set-up.

For more a detailed explanation of the steps involved, please refer to the "Atmark-Dist Developers Guide."

The operations in the following examples should be carried out under the home directory (~ /).



The development process involves working with basic libraries, applications and system configuration files. While all files are altered only under the working directory, in order to ensure the PC operating system is not inadvertently damaged from any mistakes made during development please perform all work as a **general user** and not a root user.

7.1. Building Kernel and Userland Images

The following explains how to make kernel and userland images from the source code based distribution Atmark-Dist and the Linux kernel source code.

7.1.1. Preparing Source Code

The first step is to obtain the source code. The Atmark-Dist and Linux kernel source code is needed to create images. The Aerial driver is needed when using Armadillo-WLAN Module (AWL12) and the AWL13 driver is needed when using Armadillo-WLAN Module (AWL13).

7.1.1.1. Atmark-Dist

Extract the contents of `atmark-dist-[version].tar.gz` stored in the Atmark-Dist source archive directory (`source/dist`) on the included DVD to the work directory.

After extracting the files, register the kernel source in Atmark-Dist. Perform all operations as shown in Figure 7.1, "Source Code Preparation (Atmark-Dist)".

```
[PC ~]$tar zxvf linux-[version].tar.gz
[PC ~]$ ls
atmark-dist-[version].tar.gz  atmark-dist-[version]
[PC ~]$ ln -s atmark-dist-[version] atmark-dist
```

Figure 7.1. Source Code Preparation (Atmark-Dist)

7.1.1.2. Linux Kernel

Extract the contents of `linux-[version].tar.gz` stored in the Atmark-Dist kernel source archive directory (`source/kernel`) on the included DVD to the work directory.

After extracting the files, register the kernel source in Atmark-Dist by creating a symbolic link. Perform all operations as shown in Figure 7.2, “Source Code Preparation (Linux Kernel)”.

```
[PC ~]$tar zxvf linux-[version].tar.gz
[PC ~]$ ls
atmark-dist-[version].tar.gz  atmark-dist-[version]
linux-[version].tar.gz  linux-[version]
[PC ~]$ cd atmark-dist
[PC ~/atmark-dist]$ ln -s ../linux-[version] linux-2.6.x
```

Figure 7.2. Source Code Preparation (Linux Kernel)

7.1.1.3. Aerial Driver

The following steps are only required when using Armadillo-WLAN Module (AWL12).

To create an image that uses AWL12, along with the kernel source the AWL12 device driver (Aerial driver) must also be registered in Atmark-Dist.

Extract the contents of `aerial-[version].tar.gz` stored in the Aerial driver source archive directory (`source/awlan_driver`) on the included DVD to the work directory.

After extracting the files, register the Aerial driver source in Atmark-Dist by creating a symbolic link. Perform all operations as shown in Figure 7.3, “Source Code Preparation (Aerial Driver)”.

```
[PC ~]$tar zxvf aerial-[version].tar.gz
[PC ~]$ ls
atmark-dist-[version].tar.gz  atmark-dist-[version]
linux-[version].tar.gz  linux-[version]
aerial-[version].tar.gz  aerial-[version]
[PC ~]$ cd atmark-dist
[PC ~/atmark-dist]$ ln -s ../aerial-[version] aerial
```

Figure 7.3. Source Code Preparation (Aerial Driver)

7.1.1.4. AWL13 Driver

The following steps are only required when using Armadillo-WLAN Module (AWL13).

To create an image that uses AWL13, along with the kernel source the AWL13 device driver (AWL13 driver) must also be registered in Atmark-Dist.

Extract the contents of `awl13-[version].tar.gz` stored in the AWL13 driver source archive directory (`source/awlan_driver`) on the included DVD to the work directory.

After extracting the files, register the AWL13 driver source in Atmark-Dist by creating a symbolic link. Perform all operations as shown in Figure 7.4, “Source Code Preparation (AWL13 Driver)”.

```
[PC ~]$tar zxvf awl13-[version].tar.gz
[PC ~]$ ls
atmark-dist-[version].tar.gz  atmark-dist-[version]
linux-[version].tar.gz      linux-[version]
awl13-[version].tar.gz      awl13-[version]
[PC ~]$ cd atmark-dist
[PC ~/atmark-dist]$ ln -s ../awl13-[version] awl13
```

Figure 7.4. Source Code Preparation (AWL13 Driver)

7.1.2. Applying Default Configuration

Atmark-Dist should then be configured for the target Armadillo. As the Linux kernel is managed by Atmark-Dist, once Atmark-Dist has been configured appropriately the correct kernel configuration will be applied automatically.

Start the configuration by entering the commands shown in the following example^[1].

```
[PC ~/atmark-dist]$ make config
```

First, you will be asked to select a vendor. Please enter "AtmarkTechno".

```
[PC ~/atmark-dist]$ make config
config/mkconfig > config.in
#
# No defaults found
#
*
* Vendor/Product Selection
*
*
* Select the Vendor you wish to target
*
Vendor (3com, ADI, Akizuki, Apple, Arcturus, Arnewsh, AtmarkTechno, Atmel, Avnet,
Cirrus, Cogent, Conexant, Cwlinux, CyberGuard, Cytek, Exys, Feith, Future, GDB,
Hitachi, Imt, Insight, Intel, KendinMicrel, LEOX, Mecel, Midas, Motorola, NEC,
NetSilicon, Netburner, Nintendo, OPENcores, Promise, SNEHA, SSV, SWARM, Samsung,
SecureEdge, Signal, SnapGear, Soekris, Sony, StrawberryLinux, TI, TeleIP,
Triscend, Via, Weiss, Xilinx, senTec) [SnapGear] (NEW) AtmarkTechno
```

Next, you will be asked to select a product name. Please select the appropriate product name from Table 7.1, "Product Name List".

Table 7.1. Product Name List

Product	Product Name	Notes
Armadillo-420 Basic Model	Armadillo-420	Factory Image
Armadillo-420 WLAN Model (AWL12 Compatible)	Armadillo-420.WLAN-AWL12	Factory Image
Armadillo-420 WLAN Model (AWL13 Compatible)	Armadillo-420.WLAN-AWL13	Factory Image
Armadillo-440 LCD Model	Armadillo-440	Factory Image
Armadillo-460 Basic Model	Armadillo-460	Factory Image

^[1]While this explains how to configure Atmark-Dist using the command line, it is also possible to use a menu system. For more details on using the menu system, please refer to the "Atmark-Dist Developers Guide."

Armadillo-440 is used in the example below.

```
* * Select the Product you wish to target * AtmarkTechno Products
(Armadillo-210.Base, Armadillo-210.Recover, Armadillo-220.Base,
Armadillo-220.Recover, Armadillo-230.Base, Armadillo-230.Recover,
Armadillo-240.Base, Armadillo-240.Recover, Armadillo-300, Armadillo-420,
Armadillo-420.WLAN-AWL12, Armadillo.WLAN-AWL13, Armadillo-440, Armadillo-460,
Armadillo-500, Armadillo-500-FX.base, Armadillo-500-FX.dev, Armadillo-9,
Armadillo-9.PCMCIA, SUZAKU-V.SZ310, SUZAKU-V.SZ310-SID, SUZAKU-V.SZ310-SIL,
SUZAKU-V.SZ310-SIL-GPIO, SUZAKU-V.SZ410, SUZAKU-V.SZ410-SID, SUZAKU-V.SZ410-SIL,
SUZAKU-V.SZ410-SIL-GPIO, SUZAKU-V.SZ410-SIV) [Armadillo-210.Base]
(NEW) Armadillo-440
```

You will then be asked to select the development environment. Please enter "default". The armel (EABI) development environment will be used for the Armadillo-400 Series when "default" is selected.

```
* * Kernel/Library/Defaults Selection * * * Kernel is linux-2.6.x * Cross-dev
(default, arm-vfp, arm, armel, armmommu, common, h8300, host, i386, i960,
m68knommu, microblaze, mips, powerpc, sh) [default] (NEW) default
```

Next, enter "None" for which libc to build. By selecting "None", the libc already installed in the development environment (glibc) will be used.

```
Libc Version (None, glibc, uC-libc, uClibc) [uClibc] (NEW) None
```

Enter "y" (yes) when asked whether or not to default all settings.

```
Default all settings (lose changes) (CONFIG_DEFAULTS_OVERRIDE) [N/y/?] (NEW) y
```

Enter "n" (no) for the final three questions.

```
Customize Kernel Settings (CONFIG_DEFAULTS_KERNEL) [N/y/?] n
Customize Vendor/User Settings (CONFIG_DEFAULTS_VENDOR) [N/y/?] n
Update Default Vendor Settings (CONFIG_DEFAULTS_VENDOR_UPDATE) [N/y/?] n
```

Once all options have been entered, the configuration will be saved to the build system and you will be returned to the prompt.

7.1.3. Building

To start the build, execute the command shown in Figure 7.5, "Atmark-Dist Build" in the atmark-dist directory. Once the build completes, image files will have been created under the atmark-dist/images directory. romfs.img is the userland image and linux.bin is the kernel image. romfs.img.gz and linux.bin.gz are compressed versions of these files.

```
[PC ~/atmark-dist]$ make
:
:
[PC ~/atmark-dist]$ ls images
linux.bin linux.bin.gz romfs.img romfs.img.gz
```

Figure 7.5. Atmark-Dist Build

The created images can be written to the target Armadillo by following the steps in Chapter 6, Rewriting Flash Memory. The compressed images are normally used as they use the least amount of flash memory.

7.1.4. Customizing Images

Atmark-Dist includes a variety of applications and libraries which can be included or removed from an image according to the chosen configuration. It is also possible to alter the kernel configuration.

Use the `make menuconfig` command to change the Atmark-Dist configuration.

```
[PC ~/atmark-dist]$ make menuconfig
```

Figure 7.6. Atmark-Dist Configuration

After executing the `make menuconfig` command, the Main Menu screen as shown in Figure 7.7, “menuconfig: Main Menu” is displayed.

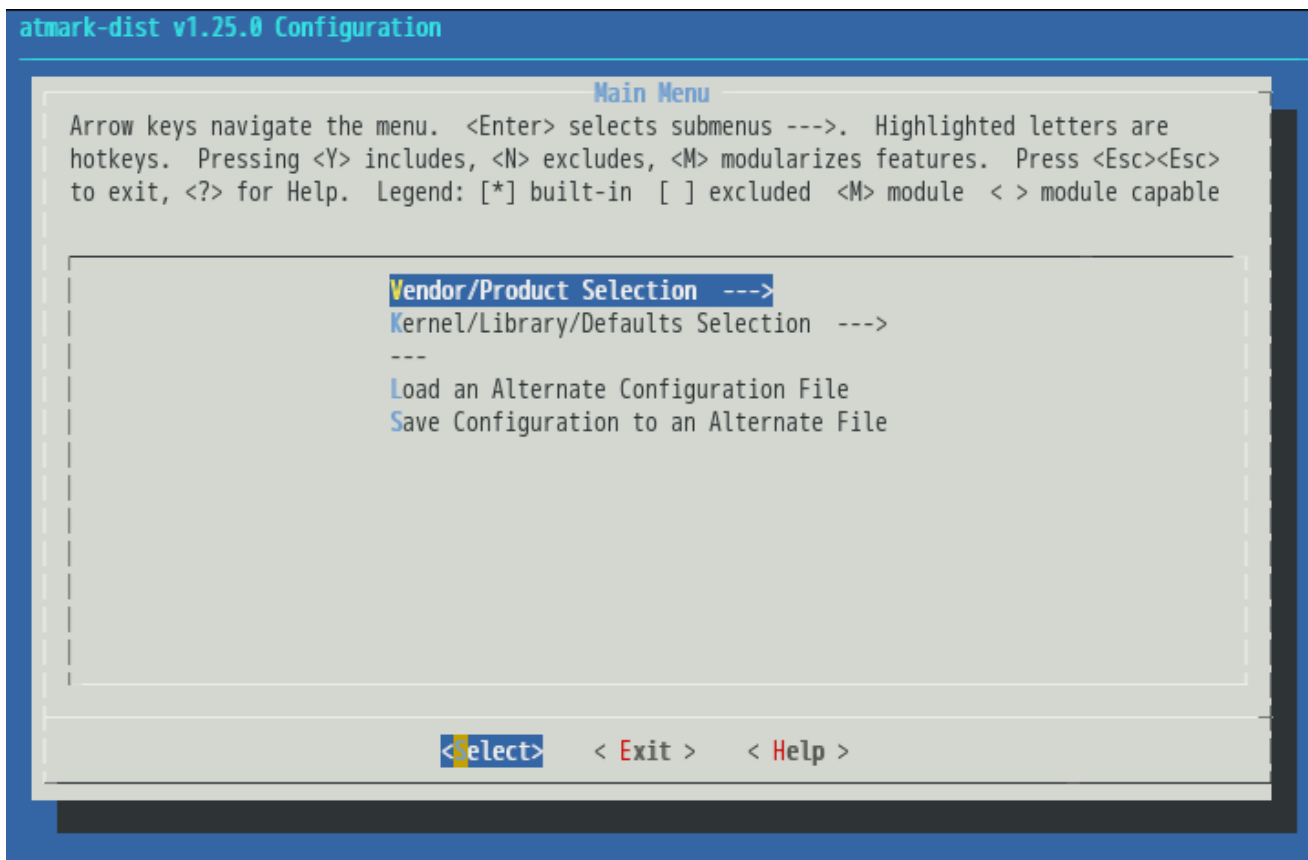


Figure 7.7. menuconfig: Main Menu

Use the up and down arrow keys to highlight **Kernel/Library/Defaults Selection** ---> and press enter. The Kernel/Library/Defaults Selection screen will be displayed.

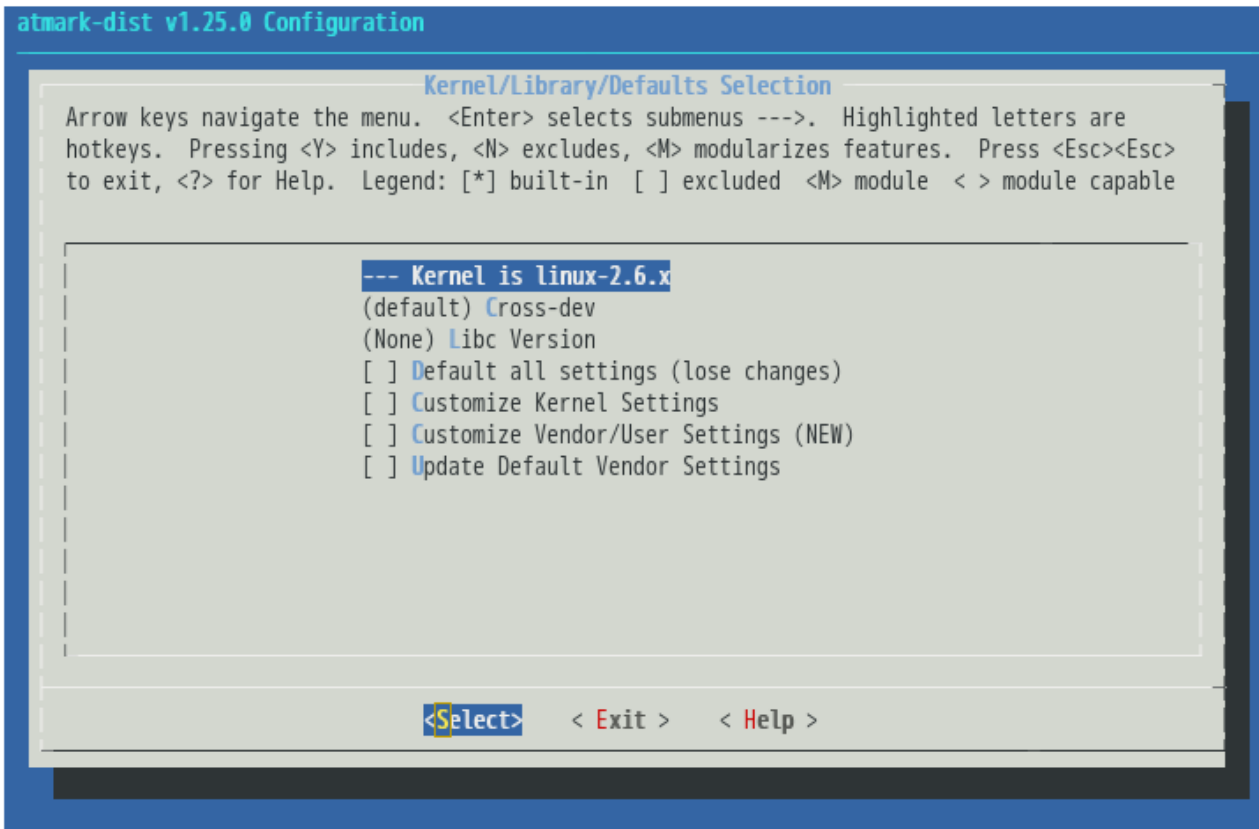


Figure 7.8. menuconfig: Kernel/Library/Defaults Selection

To change the kernel configuration, select **Customize Kernel Settings**. To change what applications or libraries are included in the userland, select **Customize Vendor/User Settings**. Each option can be selected by using the up and down arrow keys to highlight the relevant option and then pressing the space bar so that a "*" mark appears before it.

After making the selections, use the left and right arrow keys to highlight **Exit** and press Enter. You will be taken back to the Main Menu screen.

Again in the Main Menu screen highlight **Exit** and press Enter. The **Do you wish to save your new kernel configuration?** confirmation screen will then appear. Highlight **Yes** and press Enter.

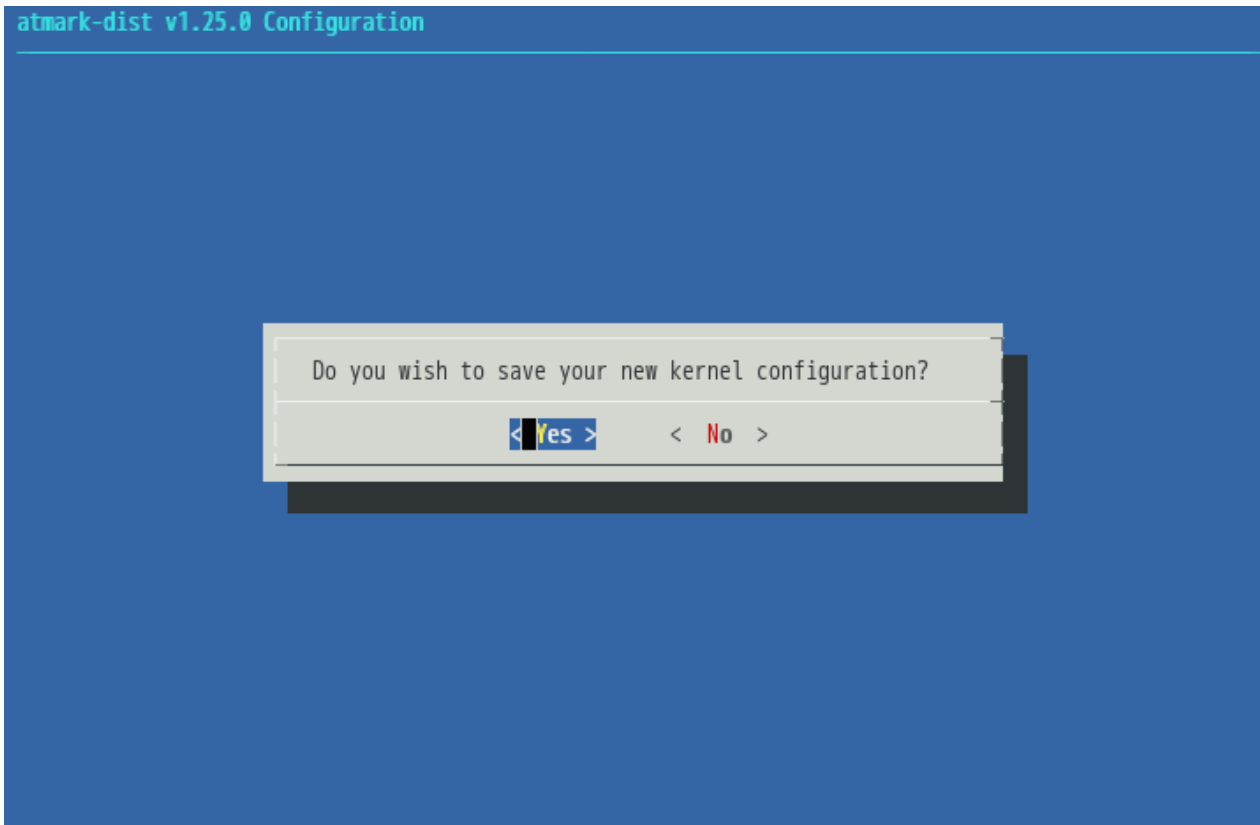


Figure 7.9. menuconfig: Do you wish to save your new kernel configuration?

If you had selected the **Customize Kernel Settings** option, the Linux Kernel Configuration screen will appear. Changes to the kernel configuration can be made here. After making the required changes, highlight **Exit** on the Linux Kernel Configuration screen and press Enter to exit.

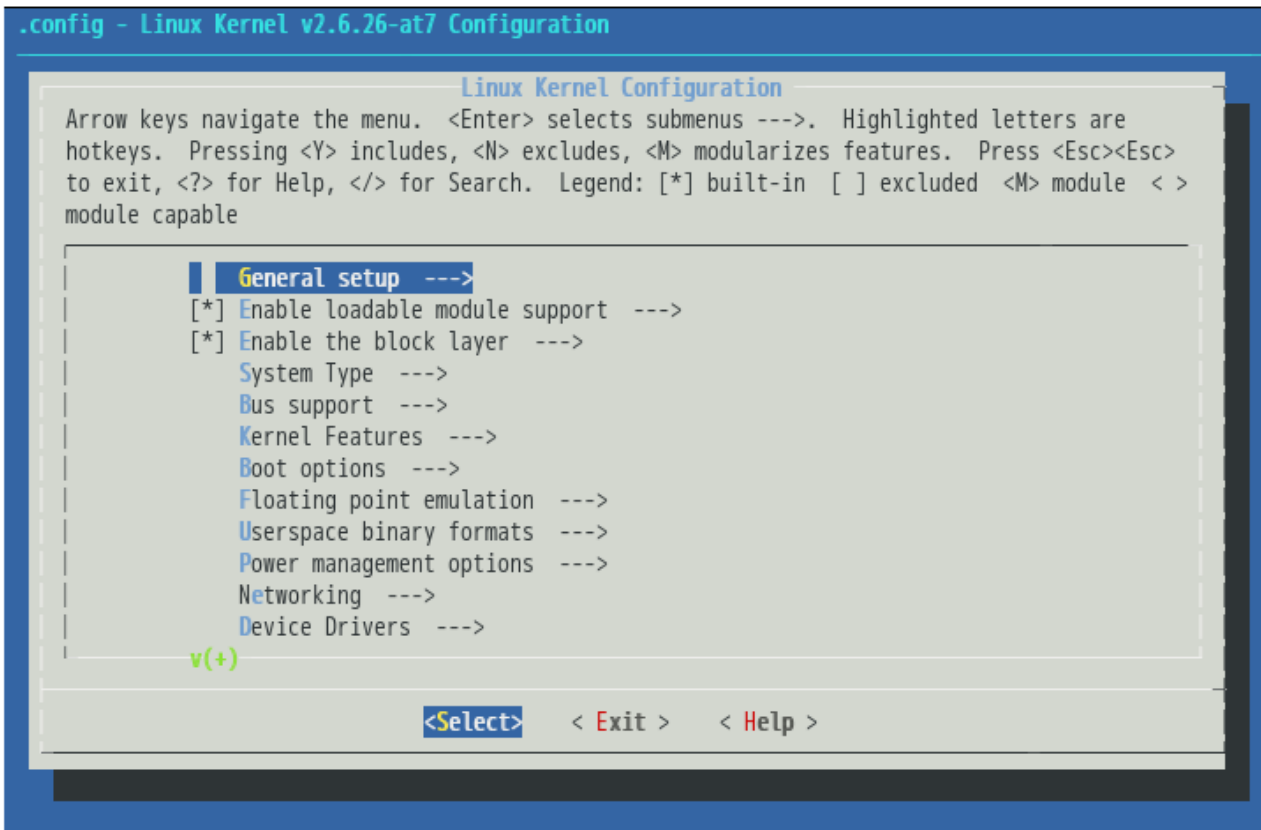


Figure 7.10. menuconfig: Linux Kernel Configuration



Configuration Changes For Using AWL12 And AWL13

After selecting an Armadillo-420/440/460 product and when using Armadillo-WLAN (AWL12 or AWL13), enable the following kernel configuration options.

```

System Type --->
  Freescale MXC Implementations --->
    MX25 Options --->
      Device Options --->
        [*] Enable eSDHC2
    Armadillo-400 Board options --->
      [*] Enable SDHC2 at CON9
      [*] Enable PWREN for SDHC2 at CON9_1
  Device Drivers --->
    Voltage and Current regulators --->
      <*> Voltage and Current Regulator Support
    
```

Figure 7.11. Enabling AWLAN

If you had selected the **Customize Vendor/User Settings** option, the Userland Configuration screen will then appear. Changes to what applications are libraries are included in the userland can be made here. After making the required changes, highlight **Exit** on the Userland Configuration screen and press Enter to exit.

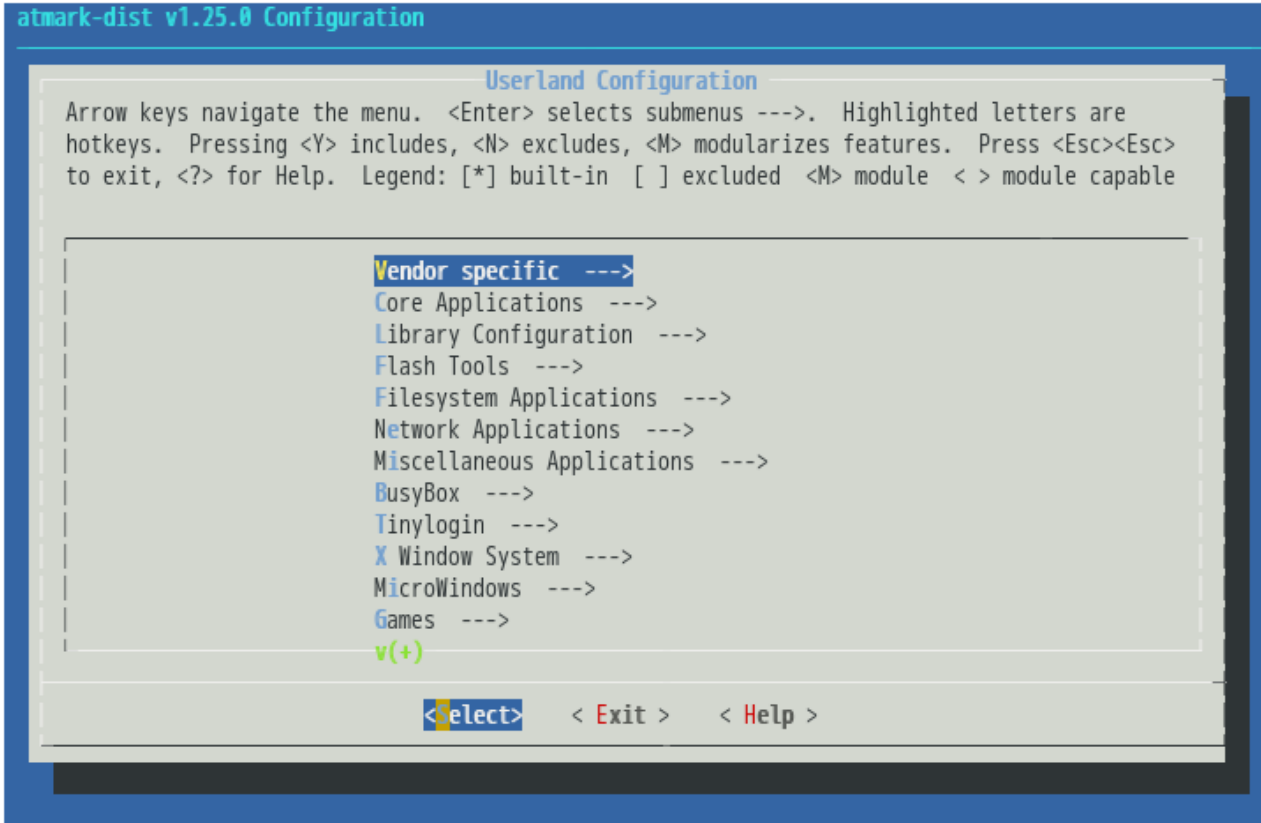


Figure 7.12. menuconfig: Userland Configuration

The **Do you wish to save your new kernel configuration?** confirmation screen will then appear once again. Highlight Yes and press Enter.

This completes the configuration process.

For more details on the using **make menuconfig** for configuration, please refer to the "Atmark-Dist Developers Guide".

After the configuration process, use the **make** command in the same way described in Section 7.1.3, "Building" to create images based off the new configuration.

7.1.5. Adding an Application to the Userland Image

The following explains how to add files such as an original application not included in Atmark-Dist to the userland created in Section 7.1, "Building Kernel and Userland Images".

The following example assumes that an original application has be built with the Out-Of-Tree method^[2] and has the filename `~/sample/hello`.

^[2]For information on Out-Of-Tree compiles, please refer to "Atmark-Dist Developers Guide".

In Atmark-Dist, the files to be included in the userland image are stored under the romfs directory. It is possible to create a userland image which includes the original application by adding the application to this directory and then executing the **make image** command.

```
[PC ~/atmark-dist]$ cp ~/sample/hello romfs/bin/
[PC ~/atmark-dist]$ make image
: : [PC ~/atmark-dist]$ ls images
linux.bin linux.bin.gz romfs.img romfs.img.gz
```

Figure 7.13. Userland Image Customization

The hello application is now installed under the /bin directory in the newly created userland image.

7.2. Building Bootloader Images

The following explains how to build a Hermit-At bootloader image from source code. The explanation applies to versions v2.0.0 or later of the source code.

7.2.1. Preparing Source Code

Extract hermit-at-[version]-source.tar.gz from the source/bootloader directory on the included DVD to the working directory. Complete the operation as shown in Figure 7.14, “Hermit-At Source Archive Extraction”.

```
[PC ~]$ tar zxvf hermit-at-[version]-source.tar.gz
[PC ~]$ ln -s hermit-at-[version] hermit-at
[PC ~]$ cd hermit-at
[PC ~/hermit-at]$
```

Figure 7.14. Hermit-At Source Archive Extraction

7.2.2. Building

In versions v2.0.0 and later of Hermit-At, default configurations for each product have been prepared as defconfig files. To build an image with the same content as the factory image, execute the commands shown in Figure 7.15, “Hermit-At Build Example”.

```
[PC ~/hermit-at]$ make armadillo4x0_defconfig
[PC ~/hermit-at]$ make
:
:
[PC ~/hermit-at]$ ls src/target/armadillo4x0/*.bin
loader-armadillo4x0-[version].bin
```

Figure 7.15. Hermit-At Build Example

The loader-armadillo4x0-[version].bin file is the newly created bootloader image.

Versions v2.0.0 and later of Hermit-At also support configuration with the **make menuconfig** command in the same way as Atmark-Dist. Commands can be added or removed and default behavior altered with this configuration method.

Chapter 8. Kernel and Userland Placement

On the Armadillo-400 Series, by default the kernel and userland images are stored in flash memory and loaded to RAM by the bootloader before the kernel is booted.

It is also possible to have the kernel and userland images loaded from places other than flash memory on the Armadillo-400 Series.

This chapter explains how to load images from other places as well as the boot options required to do so.

8.1. Loading from a TFTP Server

The tftpboot function of the Hermit-At bootloader can download kernel and userland image files from a TFTP server, load the images to RAM and then boot them.

As images are booted without first writing them to flash memory, the tftpboot function can help development efficiency during stages when the images are updated regularly.

8.1.1. File Placement

Place the kernel and userland images in the root directory of the TFTP server.



A TFTP server (atftpd) is started by default on ATDE v3.0 and later. Files placed in the `/var/lib/tftpboot/` directory will be accessible via TFTP.

8.1.2. Boot Options

Set the jumpers on the target Armadillo appropriately for maintenance mode and power on the board.

Using the serial console software on the work PC, enter the following command^[1].

```
hermit> setbootdevice tftp [Armadillo IP address] [tftp server IP address]
        --kernel=kernel_image_file_name --userland=userland_image_file_name
```

Figure 8.1. tftpboot Command

Either one or both of the kernel and userland images may be specified.

Where the TFTP server has the IP address 192.168.10.1, the Armadillo has the IP address 192.168.10.10, the kernel image filename is `linux.bin.gz` and the userland image filename is `romfs.img.gz`, the command^[2] will be as shown below.

^[1]The command is only shown as multiple lines due to formatting constraints and should be entered as one line.

^[2]The command is only shown as multiple lines due to formatting constraints and should be entered as one line.

```
hermit> setbootdevice tftp 192.168.10.10 192.168.10.1  
--kernel=linux.bin.gz --userland=romfs.img.gz
```

Figure 8.2. tftpboot Command Example

When the TFTP configuration is set with the **setbootdevice** command, the settings are saved and the kernel and userland images will be loaded from the TFTP server on all future boots.

8.2. Loading from Storage

The kernel image can be loaded from a microSD/SD card and the userland root filesystem can be stored on either a microSD/SD card or USB memory on the Armadillo-400 Series.

The following explains how to load a kernel image and root filesystem both stored on a microSD/SD card.

In this example one partition is created on a microSD/SD card and formatted as an EXT3 filesystem. A root filesystem is then created there and the kernel image placed under the `/boot/` directory. The device from which the kernel image will be loaded is specified with Hermit-At's boot options. The placement of the root filesystem is specified with kernel parameters.



The steps described here apply when using versions 2.0.3 and later of the Hermit-At bootloader. As versions 2.0.2 and earlier are not compatible with EXT3 formatted boot partitions, these steps will not be usable.

If version 2.0.2 or earlier of the Hermit-At bootloader must be used for some reason, please refer to the explanation in version 1.2.0 of the Armadillo-400 Series Software Manual.

8.2.1. Partitioning

First, make one primary partition on the microSD/SD card.

Insert the microSD/SD card into the slot^[3] and create the partition as shown in Figure 8.3, "Partitioning Procedure".

^[3]The microSD slot on the Armadillo-420/440 is a lockable type. For information on inserting and removing the microSD card, please refer to the "Armadillo-400 Series Hardware Manual".

```

[armadillo ~]# fdisk /dev/mmcblk0
The number of cylinders for this disk is set to 124277.
There is nothing wrong with that, but this is larger than 1024,
and could in certain setups cause problems with:
1) software that runs at boot time (e.g., old versions of LILO)
2) booting and partitioning software from other OSs
   (e.g., DOS FDISK, OS/2 FDISK)

Command (m for help): d ❶
Selected partition 1

Command (m for help): n ❷
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4): 1 ❸
First cylinder (1-124277, default 1):
Using default value 1
Last cylinder or +size or +sizeM or +sizeK (1-124277, default 124277): ❹
Using default value 124277


Command (m for help): w ❺
The partition table has been altered!

Calling ioctl() to re-read partition table.
mmcblk0: p1
mmcblk0: p1
Syncing disks.

[armadillo ~]#
    
```

Figure 8.3. Partitioning Procedure

- ❶ First, delete a pre-existing partition. If there is more than one pre-existing partition, please delete them all.
- ❷ Make the new primary partition on the microSD card.
- ❸ Press enter to use the default value (1) for the first cylinder.
- ❹ Press enter to use the default value (124277) for the last cylinder also.
- ❺ Write the changes to the microSD/SD card.



As the number of cylinders depends on the specifications of the microSD/SD card being used, the number may not be the same as that displayed in the procedure above.

8.2.2. Creating Filesystems

Next, format the partition as an EXT3 filesystem as shown in Figure 8.4, “Filesystem Creation Procedure”.

```
[armadillo ~]# mke2fs -j /dev/mmcblk0p1
mke2fs 1.25 (20-Sep-2001)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
497984 inodes, 994220 blocks
49711 blocks (5%) reserved for the super user
First data block=0
31 block groups
32768 blocks per group, 32768 fragments per group
16064 inodes per group
Superblock backups stored on blocks:
    32768, 98304, 163840, 229376, 294912, 819200, 884736

Writing inode tables: done
Creating journal (8192 blocks): done
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 35 mounts or
180.00 days, whichever comes first.  Use tune2fs -c or -i to override.
```

Figure 8.4. Filesystem Creation Procedure

8.2.3. Kernel Image Placement

When booting from a microSD/SD card, the kernel image must be stored in the /boot directory on the boot partition. Both uncompressed kernel images (Image, linux.bin) and compressed images (Image.gz, linux.bin.gz) are supported.

In the following example the **wget** command is used to obtain the kernel image. The URL that should be specified with the **wget** command differs depending on the product. Please refer to the table below for the correct URL.

Table 8.1. Kernel Image Download URLs

Product	URL
Armadillo-420	http://download.atmark-techno.com/armadillo-420/image/linux-a400-[version].bin.gz
Armadillo-440	http://download.atmark-techno.com/armadillo-440/image/linux-a400-[version].bin.gz
Armadillo-460	http://download.atmark-techno.com/armadillo-460/image/linux-a460-[version].bin.gz

The following example shows the kernel placement for Armadillo-440.

```
[armadillo ~]# mount /dev/mmcblk0p1 /mnt/
[armadillo ~]# mkdir /mnt/boot
[armadillo ~]# cd /mnt/boot
[armadillo /mnt/boot]# wget http://download.atmark-techno.com/armadillo-440/image/
linux-a400-[version].bin.gz
[armadillo /mnt/boot]# mv linux-a400-[version].bin.gz /mnt/boot/linux.bin.gz
[armadillo /mnt/boot]# cd
[armadillo ~]# umount /mnt
```

Figure 8.5. Kernel Image Placement

8.2.4. Creating a Root Filesystem

The following explains how to create a root filesystem on a microSD/SD card.

Either Debian/GNU Linux or a filesystem created with Atmark-Dist can be used for the root filesystem.

8.2.4.1. Installing Debian GNU/Linux

To install Debian GNU/Linux, first obtain the archive files from either the debian directory on the included DVD or from the download site. These are files from a standard Debian GNU/Linux install which have been divided up and compressed into a number of archives. The installation can be performed by just extracting the archives to the root filesystem.



A partition with at least 1GB of free space is required when installing Debian GNU/Linux as the root filesystem.

In the following example the wget command is used to obtain the debian archives. The URLs that should be specified with the wget command differ depending on the product. Please refer to the table below for the correct URLs.

Table 8.2. Debian Archive Download URLs

Product	URL
All Armadillo-400 Series Models	http://download.atmark-techno.com/armadillo-4x0/debian/debian-lenny-armel-#.tgz ^[a]
	http://download.atmark-techno.com/armadillo-4x0/debian/debian-lenny-armel-a4x0.tgz

^[a]Note: the "#" refers to the numbers 1 to 5.

```
[armadillo ~]# mount /dev/mmcb1k0p1 /mnt/
[armadillo ~]# mkdir tmp
[armadillo ~]# mount -t ramfs ramfs tmp
[armadillo ~]# cd tmp
[armadillo ~/tmp]#for N in 1 2 3 4 5 a4x0; do
> wget http://download.atmark-techno.com/armadillo-4x0/debian/debian-lenny-armel-  
{N}.tgz;
> gzip -cd debian-lenny-armel- $\{N\}$ .tgz | (cd /mnt; tar xf -);
> sync;
> rm -f debian-lenny-armel- $\{N\}$ .tgz;
> done
[armadillo ~/tmp]# cd
[armadillo ~]# umount tmp
[armadillo ~]# rmdir tmp
[armadillo ~]# umount /mnt
```

Figure 8.6. Root Filesystem Creation with Debian Archives

8.2.4.2. Using an Atmark-Dist Image

The following explains how to create a root filesystem on a microSD/SD card by using a root filesystem produced with Atmark-Dist. It is possible to use a microSD/SD with a smaller storage capacity than when installing Debian.

In the following example the wget command is used to obtain the initrd image containing the root filesystem produced with Atmark-Dist. The URL that should be specified with the wget command differs depending on the product. Please refer to the table below for the correct URL.

Table 8.3. Atmark-Dist Image Download URL

Product	URL
Armadillo-420	http://download.atmark-techno.com/armadillo-420/image/romfs-a420-[version].img.gz

Product	URL
Armadillo-440	http://download.atmark-techno.com/armadillo-440/image/romfs-a440-[version].img.gz
Armadillo-460	http://download.atmark-techno.com/armadillo-460/image/romfs-a460-[version].img.gz

```
[armadillo ~]# mount /dev/mmcblk0p1 /mnt/
[armadillo ~]# mkdir tmp
[armadillo ~]# mkdir romfs
[armadillo ~]# mount -t ramfs ramfs tmp
[armadillo ~]# wget http://download.atmark-techno.com/armadillo-440/image/romfs-
a440-[version].img.gz -P tmp
[armadillo ~]# gzip -d tmp/romfs-a440-[version].img.gz
[armadillo ~]# mount -o loop tmp/romfs-a440-[version].img romfs/
[armadillo ~]# (cd romfs/; tar cf - *) | (cd /mnt; tar xf -)
[armadillo ~]# sync
[armadillo ~]# umount romfs
[armadillo ~]# rmdir romfs
[armadillo ~]# umount tmp
[armadillo ~]# rmdir tmp
[armadillo ~]# umount /mnt
```

Figure 8.7. Root Filesystem Creation with Atmark-Dist Image

As the /etc/fstab configuration is set for flash memory in the Atmark-Dist image it must be altered for use with a microSD/SD.

```
[armadillo ~]# mount /dev/mmcblk0p1 /mnt/
[armadillo ~]# vi /mnt/etc/fstab
/dev/mmcblk0p1      /          ext3    defaults      0 1
proc                /proc      proc    defaults      0 0
usbfs               /proc/bus/usb  usbfs  defaults      0 0
sysfs               /sys       sysfs   defaults      0 0
[armadillo ~]# umount /mnt
```

Figure 8.8. fstab Alter Example

8.2.5. Boot Device and Kernel Parameter Settings

The place from which the kernel image will be loaded is selected with Hermit-At's boot device configuration. The placement of the root filesystem is specified with kernel parameters.

Set the jumpers for maintenance mode and reboot the board.

Execute the command shown in Figure 8.9, “Boot Device Designation” in order to boot from a kernel image stored in the first partition on a microSD/SD card.

```
hermit> setbootdevice mmcblk0p1
```

Figure 8.9. Boot Device Designation

Execute the command shown in Figure 8.10, “Root Filesystem Designation Example” in order to use a root filesystem stored in the first partition on a microSD/SD card.

```
hermit> setenv console=ttymxc1 root=/dev/mmcblk0p1 noinitrd rootwait
```

Figure 8.10. Root Filesystem Designation Example

For information on returning boot device and kernel parameter configuration to their initial states refer to Section 6.7, “Restoring Bootloader Parameters to Factory State”.

Chapter 9. Linux Kernel Device Driver Specifications

This chapter describes the specifications of the Linux kernel device drivers unique to the Armadillo-400 Series.

It is possible to use a wide range of functionality that is not enabled by default on the Armadillo-400 Series by altering the kernel configuration.

In order to use device drivers not enabled by default on the Armadillo-400 Series, the kernel must be configured by following the steps below.

1. Select which pins to assign the function to with board options.

When using make menuconfig, board options can be altered under **System Type -> Freescale MXC Implementations -> MX25 Options -> Armadillo-400 Board options** in the Linux Kernel Configuration.

The board options are designed so that multiple functions cannot be assigned to the same pins. Pins that do not have any specific function assigned to them will be configured as GPIO.

2. Enable the host (master) device driver.
3. If required, enable the slave device driver.
4. If required, add device information to `linux-2.6.26-at/arch/arm/mach-mx25/armadillo400.c`.

For information on altering the kernel configuration, please refer to Section 7.1.4, “Customizing Images”.

9.1. UART

The i.MX25 processor contains five UART modules: UART1 to UART5. The UART available on the Armadillo-400 Series in their default state are shown in Table 9.1, “Default Available UART”.

Table 9.1. Default Available UART

Serial Interface	Module Used	Part Number	Notes
Serial Interface 1	UART2	CON3	None
Serial Interface 2	UART3	CON9	None
Serial Interface 3	UART5	CON9	None
Serial Interface 4	UART4	CON19 ^[a]	Armadillo-460 only

^[a]Signal line shared with CON11

UART3 and UART4 can be assigned to CON11 by altering the kernel configuration. When using UART9 or UART11, CTS/RTS hardware flow control can be selectively enabled or disabled.

The UART driver has the following functionality.

- 7/8 bit send and receive
- 1/2 stop bits
- None/Odd/Even parity

- XON/XOFF software flow control
- CTS/RTS hardware flow control
- Modem line control
- Standard Linux serial API
- Max baud rate: 230.4Kbps (Serial Interface 1, 4) / 4Mbps^[1] (Serial Interface 2, 3)

Each serial interface and their corresponding device files are shown in Table 9.2, “Serial Interface Device Files”.

Table 9.2. Serial Interface Device Files

Serial Interface	Device File	Module Used
Serial Interface 1	/dev/ttymxcl	UART2
Serial Interface 2	/dev/ttymxc2	UART3
Serial Interface 3	/dev/ttymxc4	UART5
Serial Interface 4	/dev/ttymxc3	UART4

Kernel configuration options related to the UART functionality and common to all Armadillo-400 Series models are shown in Table 9.3, “UART Configuration”, while those specific to Armadillo-460 are shown in Table 9.4, “UART Configuration (Armadillo-460 Specific)”.

Table 9.3. UART Configuration

Kernel Configuration Options	Default	Description
SERIAL_MXC	y	Enables the i.MX serial driver
SERIAL_MXC_CONSOLE	y	Enables the system console for the i.MX serial driver
ARMADILLO400_UART3_CON9	y	Enables UART3 on CON9 Uses CON9_3 for UART3_RXD and CON9_5 for UART3_TXD
ARMADILLO400_UART3_HW_FLOW_CON9	n	Enables hardware flow control for UART3 on CON9 Uses CON9_11 for UART3_RTS and CON9_13 for UART3_CTS Depends on ARMADILLO400_UART3_CON9
ARMADILLO400_UART3_CON11	n	Enables UART3 on CON11 Uses CON11_40 for UART3_RXD and CON11_41 for UART3_TXD Cannot be used with ARMADILLO400_UART3_CON9
ARMADILLO400_UART3_HW_FLOW_CON11	n	Enables hardware flow control for UART3 on CON11 Uses CON11_42 for UART3_RTS and CON11_43 for UART3_CTS Depends on ARMADILLO400_UART3_CON11
ARMADILLO400_UART4_CON11	n	Enables UART4 on CON11 Uses CON11_44 for UART4_RXD and CON11_45 for UART4_TXD
ARMADILLO400_UART4_HW_FLOW_CON11	n	Enables hardware flow control for UART4 on CON11 Uses CON11_46 for UART4_RTS and CON11_47 for UART4_CTS Depends on ARMADILLO400_UART4_CON11
ARMADILLO400_UART5_CON9	y	Enables UART5 on CON9 Uses CON9_4 for UART5_RXD and CON9_6 for UART5_TXD
ARMADILLO400_UART5_HW_FLOW_CON9	n	Enables hardware flow control for UART5 on CON9 Uses CON9_12 for UART5_RTS and CON9_14 for UART5_CTS Depends on ARMADILLO400_UART5_CON9

Table 9.4. UART Configuration (Armadillo-460 Specific)

Kernel Configuration Options	Default	Description
ARMADILLO460_UART4_CON19	y	Enables UART4 on CON19 Uses CON19_3 for UART4_RXD and CON19_5 for UART4_TXD

^[1]When DMA transfer is enabled. DMA is not enabled by default.

Kernel Configuration Options	Default	Description
ARMADILLO460_UART4_HW_FLOW_CON19	y	Enables hardware flow control for UART4 on CON19 Uses CON19_4 for UART4_RTS and CON19_6 for UART4_CTS Depends on ARMADILLO460_UART4_CON19

9.2. Ethernet

The Armadillo-400 Series Ethernet driver has the following functionality.

- Auto-negotiation support
- Carrier detect support
- Ethtool support
 - Link status
 - 10/100Mbps speed
 - Full/Half Duplex
 - Auto-negotiation enable/disable

Kernel configuration options related to Ethernet functionality are shown in Table 9.5, “Ethernet Configuration”.

Table 9.5. Ethernet Configuration

Kernel Configuration Options	Default	Description
NETDEVICES	y	Enables Linux kernel network device support
NET_ETHERNET	y	Enables Linux kernel 10/100 Mbps Ethernet
MX25_FEC	y	Enables the i.MX25 FEC (Fast Ethernet Controller) driver

9.3. MMC/SD/SDIO Host

The i.MX25 processor incorporates two MMC/SD/SDIO host controllers (eSDHC). On the Armadillo-400 Series, by default eSDHC1 is used for the microSD/SD slot (CON1). Also, eSDHC2 can be assigned to CON9 by altering the kernel configuration.

The Armadillo-400 Series MMC/SD/SDIO host driver has the following functionality.

- 4 bit mode
- Card detect

When a card is inserted in the microSD/SD card slot, it will be registered as /dev/mmcblkN (N is 0 or 1).

Kernel configuration options related to the MMC/SD/SDIO host functionality are shown in Table 9.6, “MMC/SD/SDIO Host Controller Configuration”.

Table 9.6. MMC/SD/SDIO Host Controller Configuration

Kernel Configuration Options	Default	Description
MMC	y	Enables Linux kernel MMC/SD/SDIO card support
MMC_UNSAFE_RESUME	y	Do not probe SD/MMC cards when resuming from sleep. For details refer to Section 9.20.1.2, “Treatment of External Devices During Sleep”.
MMC_BLOCK	y	Enables Linux kernel MMC block device driver
MMC_BLOCK_BOUNCE	y	Set whether or not the MMC driver uses a bounce buffer
MMC_IMX_ESDHCI	y	Enables the i.MX25 eSDHC driver

Kernel Configuration Options	Default	Description
ARMADILLO400_SDHC2_CON9	n	Enables SDHC2 on CON9 Uses CON9_15 to CON9_24

9.4. USB 2.0 Host

The Armadillo-400 Series USB 2.0 host driver has the following functionality.

- EHCI compliant
- OTG not supported
- USB High Speed host x1
- USB Full Speed host x1

When a USB device is detected, it will be mapped to `/dev/sd*`.

Kernel configuration options related to the USB host functionality are shown in Table 9.7, “USB Host Configuration”.

Table 9.7. USB Host Configuration

Kernel Configuration Options	Default	Description
USB_SUPPORT	y	Enables Linux kernel USB support
USB	y	Enables Linux kernel USB host support
USB_EHCI_HCD	y	Enables Linux kernel EHCI (Enhanced Host Controller Interface) support
USB_EHCI_ARC	y	Enables the i.MX USB driver
USB_EHCI_ARC_H2	y	Enables i.MX USB Host2 support
USB_EHCI_ARC_H2_DELAYPROBE	n	Enables the Delayed Probe function on USB Host2
USB_EHCI_ARC_H2_WAKE_UP	n	Enables wakeup with USB Host2 ^[a]
USB_EHCI_ARC_H2_FSL_SERIAL	y	Use the on-chip Full Speed serial transceiver as the USB Host2 PHY
USB_EHCI_ARC_OTG	y	Enables i.MX OTG port support ^[b]
USB_EHCI_ARC_OTG_DELAYPROBE	n	Enables the Delayed Probe function on the OTG port
USB_EHCI_ARC_OTG_WAKE_UP	n	Enables wakeup with the OTG port ^[a]
USB_EHCI_ARC_OTG_FSL_UTMI	y	Use the on-chip High Speed UTMI transceiver as the OTG port PHY
USB_STATIC_IRAM	y	Use internal RAM for USB data transfer
USB_STATIC_IRAM_TD_SIZE	2048	Specify the internal RAM size used for USB data transfer

^[a]Not supported on the Armadillo-400 Series.

^[b]Only host functionality is supported on the Armadillo-400 Series.

9.5. Frame Buffer

The video out function on Armadillo-440/460 is implemented as a frame buffer device.

The frame buffer device driver has the following functionality.

- Double-buffer support
- Max resolution: SVGA

The default settings on the Armadillo-440 LCD Model and the Armadillo-460 Basic Model are shown below.

- Resolution: 480 x 272 pixels
- RGB 565 Color

The frame buffers and their corresponding device files are shown in Table 9.8, “Frame Buffer Device Files”.

Table 9.8. Frame Buffer Device Files

Frame Buffer	Device File
Background plane	/dev/fb0
Graphic window	/dev/fb1

Kernel configuration options related to the frame buffer are shown in Table 9.9, “Frame Buffer Configuration”.

Table 9.9. Frame Buffer Configuration

Kernel Configuration Options	Default	Description
FB	y	Enables Linux kernel frame buffer support
FB_MXC	y	Enables the i.MX25 frame buffer driver
FB_MXC_MODE_FG040360DSSWBG03	y	Sets the frame buffer video mode to that compatible with FG040360DSSWBG03. Other LCD can be supported by changing the video mode.
FB_MXC_BPP_16	y	Sets bpp to 16
MXC_SYNC_PANEL	y	Sets the frame buffer to synchronous mode
FRAMEBUFFER_CONSOLE	y	Enables the frame buffer console
LOGO	y	Enables the boot logo
LOGO_ARMADILLO_CLUT224	y	Enables the Armadillo boot logo

9.6. LED Backlight

The LED backlight functionality on Armadillo-440/460 is implemented as a backlight class. On Armadillo-440/460, backlight control is performed with general purpose PWM^[2]. For details on the PWM functionality please refer to Section 9.17, “PWM”.

Backlight control can be performed with the files under the /sys/class/backlight/pwm-backlight directory. Brightness can be adjusted with the brightness file. Write a value between 0 (off) and 255 (full brightness) to the brightness file to change the brightness. The current brightness can be obtained by reading the value from the brightness file. The backlight on/off can be controlled with the bl_power file. Write 0 to bl_power to turn the backlight off and write 1 to turn it back on.

Kernel configuration options related to the LED backlight are shown in Table 9.10, “LED Backlight Configuration”.

Table 9.10. LED Backlight Configuration

Kernel Configuration Options	Default	Description
BACKLIGHT_LCD_SUPPORT	y	Enables Linux kernel backlight and LCD support
BACKLIGHT_CLASS_DEVICE	y	Enables Linux kernel backlight class support
BACKLIGHT_PWM	y	Enables the PWM based backlight driver

9.7. Touchscreen

The touchscreen function on Armadillo-440/460 is implemented as an input device and offers an event interface to userland programs.

The events sent by the interface are shown in Table 9.11, “Touchscreen Events”.


Table 9.11. Touchscreen Events

Type	Code	Value
EV_KEY(1)	BTN_TOUCH(330)	0 or 1
EV_ABS(3)	ABS_X(0)	100 - 4000

^[2]The LCDC (Liquid Crystal Display Controller) in the i.MX25 processor on the Armadillo-400 Series boards includes backlight control functionality, but this is not used on Armadillo-440/460.

Type	Code	Value
EV_ABS(3)	ABS_Y(1)	100 - 4000
EV_ABS(3)	ABS_PRESSURE(24)	0 or 1

On the Armadillo-440 LCD Model and the Armadillo-460 Basic Model, by default the touchscreen event device is mapped to /dev/input/event1.



The event device file numbers are determined by the order the input devices are detected. Therefore, if another input device such as a USB keyboard is detected at boot time, the event device number of the touchscreen may change.

Kernel configuration options related to the touchscreen are shown in Table 9.12, “Touchscreen Configuration”.

Table 9.12. Touchscreen Configuration


Kernel Configuration Options	Default	Description
INPUT	y	Enable Linux kernel input layer support
INPUT_EVDEV	y	Enable input layer event device support
INPUT_TOUCHSCREEN	y	Enables Linux kernel touchscreen support
TOUCHSCREEN_IMX_ADC	y	Enables the i.MX touchscreen driver

9.8. Audio

The audio function on the Armadillo-400 Series is implemented as an ALSA device driver. The ALSA device driver has the following functionality.


- Playback (2ch) / Capture (1ch)
- Sampling rates: 48k, 32k, 16k, 8k Hz
- Formats: Signed 16/20/24 bit, Little-endian

The audio device can be controlled via the ALSA library (libasound2).



The Armadillo-400 Series audio driver cannot record and play sound at the same time.

On the i.MX25, it is possible to select what pins to assign the audio function to with the audio multiplex function. On the Armadillo-400 Series, the audio multiplex settings can be changed in the kernel configuration. By default on Armadillo-440 the audio multiplexing is set to use AUD5, with the audio function connected to CON11. By altering the configuration, it is possible to connect the audio function to CON9 by using AUD6.



As the AUD5 signal on CON11 is used for UART4 by default on Armadillo-460^[3], it cannot be used for the audio function. The kernel configuration must be changed in order to use audio on Armadillo-460.

^[3]CON11_44 to CON11_47 and CON19_3 to CON19_6 share the same signal lines.

Kernel configuration options related to the audio functionality are shown in Table 9.13, “Audio Configuration”.

Table 9.13. Audio Configuration

Kernel Configuration Options	Default		Description
	Armadillo-440	Armadillo-460	
SOUND	y	n	Enables Linux kernel sound card support
SND	y	n	Enables Linux kernel ALSA support
SND_SOC	y	n	Enables Linux kernel ASoC support
SND_MXC_SOC	y	n	Enables driver which provides i.MX audio functionality
SND_SOC_ARMADILLO440_WM8978	y	n	Enables driver which provides audio functionality using the WM8978 codec on the Armadillo-400 Series
ARMADILLO400_AUD5_CON11	y	n	Assign audio function to CON11 Uses CON11_42 to CON11_47
ARMADILLO400_AUD6_CON9	n	n	Assign audio function to CON9 Uses CON9_15, 17, 21, 22, 23, 24 Cannot be used with ARMADILLO400_AUD5_CON11

9.9. GPIO

The GPIO on the Armadillo-400 Series are implemented as generic GPIO.

There are two interfaces available to manipulate GPIO from userland: GPIO sysfs and an Armadillo-200 Series compatible GPIO driver. The GPIO sysfs driver is enabled by default.

Pins that do not have any other function assigned to them in the kernel configuration are all set to be GPIO.

9.9.1. GPIO sysfs

With GPIO sysfs, I/O direction and output levels can be set and input levels checked with the files under the `/sys/class/gpio/(GPIO_NAME)` directories.

The GPIO_NAME directories and their corresponding GPIO pins are shown in Table 9.14, “GPIO_NAME and GPIO Pins”.

Table 9.14. GPIO_NAME and GPIO Pins

GPIO_NAME	GPIO Pin	Initial I/O Direction	Initial Output Level
CON9_1	CON9 pin 1	Input	-
CON9_2	CON9 pin 2	Input	-
CON9_11	CON9 pin 11	Input	-
CON9_12	CON9 pin 12	Input	-
CON9_13	CON9 pin 13	Input	-
CON9_14	CON9 pin 14	Input	-
CON9_15	CON9 pin 15	Input	-
CON9_16	CON9 pin 16	Input	-
CON9_17	CON9 pin 17	Input	-
CON9_18	CON9 pin 18	Input	-
CON9_21	CON9 pin 21	Input	-
CON9_22	CON9 pin 22	Input	-
CON9_23	CON9 pin 23	Input	-
CON9_24	CON9 pin 24	Input	-
CON9_25	CON9 pin 25	Input	-
CON9_26	CON9 pin 26	Input	-
CON9_27	CON9 pin 27	Output	LOW
CON9_28	CON9 pin 28	Output	LOW

I/O direction configuration is done with the `/sys/class/gpio/(GPIO_NAME)/direction` file by writing one of the options shown in Table 9.15, “GPIO I/O Direction Configuration”. The current configuration can be obtained by reading the value from the `direction` file.

Table 9.15. GPIO I/O Direction Configuration

Configuration	Description
high	Sets the I/O direction to output and the output level to high. The output level can be obtained and set in this state.
low	Sets the I/O direction to output and the output level to low. The output level can be obtained and set in this state.
out	This is the same as setting "low".
in	Sets the I/O direction to input. The input level can be obtained in this state.

Output levels can be set and input levels checked with the `/sys/class/gpio/(GPIO_NAME)/value` file. A 0 represents a low level and 1 a high level.

Interrupt type configuration is done with the `/sys/class/gpio/(GPIO_NAME)/edge` file by writing one of the options shown in Table 9.16, “GPIO Interrupt Type Configuration”. The current configuration can be obtained by reading the value from the `edge` file.

Table 9.16. GPIO Interrupt Type Configuration

Configuration	Description
none	Interrupt detection is not performed.
falling	Falling edge interrupt detection is performed.
rising	Rising edge interrupt detection is performed.
both	Both falling edge and rising edge interrupt detection are performed.

An example of handling GPIO sysfs interrupts in the C language is shown in Figure 9.1, “GPIO sysfs Interrupt Sample Program”. When executed, the sample program sets the I/O direction of CON9_1 to input, the interrupt type to falling edge, and then waits for an interrupt. When an interrupt is detected the level of the GPIO pin at that time is displayed. The program exits after interrupts have been detected three times.

```

#include <stdio.h>
#include <unistd.h>
#include <poll.h>
#include <fcntl.h>

#define GPIO_DIR "/sys/class/gpio"
#define GPIO_NAME "CON9_1"
#define GPIO_PATH GPIO_DIR "/" GPIO_NAME "/"

int main(void)
{
    int fd;
    int i;

    fd = open(GPIO_PATH "direction", O_RDWR);
    write(fd, "in", 2);
    close(fd);

    fd = open(GPIO_PATH "edge", O_RDWR);
    write(fd, "falling", 7);
    close(fd);

    for (i=0; i < 3; i++) {
        char val;
        struct pollfd pfd;

        fd = open(GPIO_PATH "value", O_RDWR);
        read(fd, &val, 1);

        printf("waiting for interrupt..."); fflush(stdout);

        pfd.fd = fd;
        pfd.events = POLLIN;
        pfd.revents = 0;
        poll(&pfd, 1, -1);

        lseek(fd, 0, SEEK_SET);
        read(fd, &val, 1);
        close(fd);

        printf("OK (%c, %s)\n", val, val == '0' ? "Low" : "High");
        usleep(100000);
    }

    return 0;
}

```

Figure 9.1. GPIO sysfs Interrupt Sample Program

- ❶ The interrupt type is set to falling edge by writing "falling" to the edge file.
- ❷ After specifying the interrupt type, the value file is opened.
- ❸ A dummy read is performed once on the value file. Any interrupts occurring after this will be picked up in the polling.

- 4 The poll or select system call is used to wait for an interrupt to occur.
- 5 As a dummy read has been performed, in order to obtain the level of the GPIO pin after an interrupt has occurred the lseek system call must be used to return the offset to the start of the file.


Kernel configuration options related to the GPIO sysfs are shown in Table 9.17, “GPIO sysfs Configuration”.

Table 9.17. GPIO sysfs Configuration

Kernel Configuration Options	Default	Description
GPIO_SYSFS	y	Enables Linux kernel GPIO sysfs support
GPIO_SYSFS_PRIVATE_NAMING	y	Enables the GPIO sysfs alias export function

9.9.2. Armadillo-200 Series Compatible GPIO Driver

With the Armadillo-200 Series compatible GPIO driver, GPIO are manipulated by issuing ioctl function calls to the driver specific device file.




The Armadillo-200 Series compatible GPIO driver is not enabled by default. To enable the driver, in the kernel configuration enable the CONFIG_ARMADILLO2X0_GPIO option after disabling the CONFIG_GPIO_SYSFS option and then rebuild the kernel.

The GPIO names and corresponding GPIO pins for the Armadillo-200 Series compatible GPIO driver are shown in Table 9.18, “Armadillo-200 Series Compatible GPIO Driver GPIO List”.

Table 9.18. Armadillo-200 Series Compatible GPIO Driver GPIO List

GPIO Name	GPIO Pin	Initial I/O Direction	Initial Output Level
GPIO0	CON9 pin 21	Input	-
GPIO1	CON9 pin 22	Input	-
GPIO2	CON9 pin 23	Input	-
GPIO3	CON9 pin 24	Input	-
GPIO4	CON9 pin 25	Input	-
GPIO5	CON9 pin 26	Input	-
GPIO6	CON9 pin 27	Output	LOW
GPIO7	CON9 pin 28	Output	LOW
GPIO8	CON9 pin 11	Input	-
GPIO9	CON9 pin 12	Input	-
GPIO10	CON9 pin 13	Input	-
GPIO11	CON9 pin 14	Input	-
GPIO12	CON9 pin 15	Input	-
GPIO13	CON9 pin 16	Input	-
GPIO14	CON9 pin 17	Input	-
GPIO15	CON9 pin 18	Input	-



With the Armadillo-200 Series compatible GPIO driver, the GPIO names and the corresponding GPIO pin positions are the same as those on the Armadillo-200 Series. Because of this, only a subsection of the GPIO available on the Armadillo-400 Series can be used with the Armadillo-200 Series compatible GPIO driver.

The details of the device file are shown below.

Table 9.19. Armadillo-200 Series Compatible GPIO Driver Device File

Type	Major Number	Minor Number	Device File
Character device	10	185	/dev/gpio

The file descriptor of the device file is specified as the first argument to `ioctl`, and the command to manipulate the GPIO is specified as the second argument.

Table 9.20. Armadillo-200 Series Compatible GPIO Driver `ioctl` Commands

Command	Description	Third Argument Type
PARAM_SET	Configure GPIO with the state specified in the third argument	struct gpio_param
PARAM_GET	Get GPIO state by following the details specified in the third argument	struct gpio_param
INTERRUPT_WAIT	Wait for GPIO interrupt by following the details specified in the third argument	struct wait_param

The structures "struct gpio_param" and "struct wait_param" defined in `(kernel source)/include/linux/armadillo2x0_gpio.h` are used for the third argument. "struct gpio_param" can be linked with its "next" member to form a singly-linked list in order to control multiple GPIO at once. Be sure to set the "next" member of the last structure of the list to "0 (NULL)". For detailed usage information, please refer to the source code of the GPIO control application (`atmark-dist/vendors/AtmarkTechno/Armadillo-4x0.Common/gpioctrl`).

Kernel configuration options related to the Armadillo-200 Series Compatible GPIO driver are shown in Table 9.21, "Armadillo-200 Series Compatible GPIO Driver Configuration".

Table 9.21. Armadillo-200 Series Compatible GPIO Driver Configuration

Kernel Configuration Options	Default	Description
ARMADILLO2X0_GPIO	n	Enables the Armadillo-200 Series Compatible GPIO driver ^[a]

^[a]Conflicts with `GPIO_SYSFS`, so can only be selected when `GPIO_SYSFS=n`

9.10. LED

There are two LED drivers for the Armadillo-400 Series: a LED class driver and an Armadillo-200 Series compatible LED driver. The LED class driver is enabled by default.

9.10.1. LED Class

The LEDs can be controlled with the files under the `/sys/class/leds/(LED_NAME)` directory.

The `/sys/class/leds/(LED_NAME)/brightness` file is used to turn the LEDs on and off. Writing 0 to the `brightness` file turns the LED off, and writing any other number turns the LED on.

A trigger mechanism is used to control blinking with the LED class. The `mmc0`, `timer`, `heartbeat` and `default-on` triggers can be used on the Armadillo-400 Series. The trigger can be set by writing the respective character string to the `/sys/class/leds/(LED_NAME)/trigger` file. When the `mmc0` trigger is set, the LED will blink on and off in time with data accesses to the MMC/SD card. The `timer` trigger can be used to have the LED blink at a certain rate. After the `timer` trigger has been enabled, files `/sys/class/leds/(LED_NAME)/delay_on` and `/sys/class/leds/(LED_NAME)/delay_off` will be created. The length of time (msec) the LED should stay on for each cycle is written to the first file, and the time (msec) it should stay off for to the second file. When the `heartbeat` trigger is set, the LED will blink regularly while the system is running. When the `default-on` trigger is set, the LED will remain on continuously while the system is running.

The `LED_NAME` names and their respective LEDs are shown in Table 9.22, "LED List".

Table 9.22. LED List

LED_NAME	Corresponding LED	Default Trigger
red	LED3	default-on
green	LED4	default-on
yellow	LED5	None


Kernel configuration options related to the LED class are shown in Table 9.23, “LED Class Configuration”.

Table 9.23. LED Class Configuration

Kernel Configuration Options	Default	Description
NEW_LEDS	y	Enables Linux kernel LED support
LEDS_CLASS	y	Enables Linux kernel LED class support
LEDS_GPIO	y	Enables GPIO connected LED class support
LEDS_TRIGGERS	y	Enables LED class trigger support
LEDS_TRIGGER_TIMER	y	Enables timer trigger support
LEDS_TRIGGER_HEARTBEAT	y	Enables heartbeat trigger support
LEDS_TRIGGER_DEFAULT_ON	y	Enables default ON trigger support

9.10.2. Armadillo-200 Series Compatible LED Driver

With the Armadillo-200 Series compatible LED driver, LEDs are manipulated by issuing ioctl function calls to the driver specific device file.



The Armadillo-200 Series compatible LED driver is not enabled by default. To enable the driver, in the kernel configuration enable the CONFIG_ARMADILLO2X0_LED option after disabling the CONFIG_LEDS_GPIO option and then rebuild the kernel.

The details of the LED device file are shown below.

Table 9.24. LED Node

Type	Major Number	Minor Number	Device File
Character device	10	215	/dev/led

The file descriptor of the device file is specified as the first argument to ioctl, and the command to manipulate the LEDs is specified as the second argument.

Table 9.25. LED Manipulation Commands

Command	Description	Third Argument Type
LED_RED_ON	Turn LED3 (red) on	None
LED_RED_OFF	Turn LED3 (red) off	None
LED_RED_STATUS	Obtain state of LED3 (red)	Buffer to store state (1 byte min)
LED_RED_BLINKON	Start LED3 (red) blinking	None
LED_RED_BLINKOFF	Stop LED3 (red) blinking	None
LED_RED_BLINKSTATUS	Obtain blinking status of LED3 (red)	Buffer to store state (1 byte min)
LED_GREEN_ON	Turn LED4 (green) on	None
LED_GREEN_OFF	Turn LED4 (green) off	None
LED_GREEN_STATUS	Obtain status of LED4 (green)	Buffer to store state (1 byte min)
LED_GREEN_BLINKON	Start LED4 (green) blinking	None
LED_GREEN_BLINKOFF	Stop LED4 (green) blinking	None

Command	Description	Third Argument Type
LED_GREEN_BLINKSTATUS	Obtain blinking status of LED4 (green)	Buffer to store state (1 byte min)

For detailed usage information on the LED device driver, please refer to the source code of the sample LED control application (`atmark-dist/vendors/AtmarkTechno/Armadillo-440/ledctrl`).

Kernel configuration options related to the Armadillo-200 Series Compatible LED driver are shown in Table 9.26, “Armadillo-200 Series Compatible LED Driver Configuration”.

Table 9.26. Armadillo-200 Series Compatible LED Driver Configuration

Kernel Configuration Options	Default	Description
ARMADILLO2X0_LED	n	Enables the Armadillo-200 Series Compatible LED driver

9.11. Buttons

The button input function on the Armadillo-400 Series is implemented as an input device and offers an event interface to userland programs.

An on-board tact switch is available on all Armadillo-400 Series models as a button device. Also, the Armadillo-400 Series LCD Expansion Board has three buttons mounted which are available as button devices on Armadillo-440/460.


The buttons and their corresponding events are shown in Table 9.27, “Armadillo-400 Series Button Events”.

Table 9.27. Armadillo-400 Series Button Events


Buttons	Type	Code	Value
SW1	EV_KEY(1)	KEY_ENTER(28)	0 or 1
LCD_SW1 ^[a]	EV_KEY(1)	KEY_BACK(158)	0 or 1
LCD_SW2 ^[a]	EV_KEY(1)	KEY_MENU(139)	0 or 1
LCD_SW3 ^[a]	EV_KEY(1)	KEY_HOME(102)	0 or 1

^[a]Usable when the Armadillo-400 Series LCD Expansion Board is connected to Armadillo-440/460

By default the button event device is mapped to `/dev/input/event0`.



The event device file numbers are determined by the order the input devices are detected. Therefore, if another input device such as a USB keyboard is detected at boot time, the event device number of the button function may change.



As SW2 on Armadillo-460 is a reset switch it cannot be used as a button device.

Kernel configuration options related to buttons are shown in Table 9.28, “Button Configuration”.

Table 9.28. Button Configuration

Kernel Configuration Options	Default	Description
INPUT	y	Enable Linux kernel input layer support
INPUT_EVDEV	y	Enable input layer event device support
INPUT_KEYBOARD	y	Enables Linux kernel keyboard support

Kernel Configuration Options	Default	Description
KEYBOARD_GPIO	y	Enables the GPIO keyboard driver

9.12. Real-time Clock

A real-time clock (Seiko Instruments S-35390A) is included on Armadillo-460, Armadillo-400 Series LCD Expansion Board, Armadillo-400 Series RTC Option Module, and WLAN Option Module (AWL12 Compatible and AWL13 Compatible). To use a real-time clock on Armadillo-420/440, an expansion board or option module must be attached.

The real-time clock operates as an I2C slave device attached to a I2C bus. The real-time clock and I2C bus connection is shown in Table 9.29, “Real-Time Clock I2C Bus Connection”.

Table 9.29. Real-Time Clock I2C Bus Connection

RTC Equipped Board / Option Module Name	I2C Bus	Address	Priority
Armadillo-400 Series RTC Option Module	I2C2	0x30	1
Armadillo-400 Series WLAN Option Module (AWL12 Compatible)	I2C2	0x30	1
Armadillo-400 Series WLAN Option Module (AWL13 Compatible)	I2C2	0x30	1
Armadillo-400 Series LCD Expansion Board	I2C3	0x30	2
Armadillo-460	I2C-GPIO	0x30	3

Real-time clocks can be operated with either a device file or sysfs file. Device files are created as `/dev/rtcN`, and sysfs files are created under a `/sys/class/rtc/rtcN/` directory [4]. When just one real-time clock is attached, it can be operated with either the `/dev/rtc0` device file or the sysfs files under the `/sys/class/rtc/rtc0/` directory. When two or more real-time clocks are attached, they will be assigned `/dev/rtc0` then `/dev/rtc1` starting with the real-time clock with the smaller "Priority" value in Table 9.29, “Real-Time Clock I2C Bus Connection”[5]. Only `/dev/rtc0` is used in applications like `hwclock` included in `atmark-dist`.

For information on the device file interface, please refer to `linux-2.6.26-at/Documents/rtc.txt`. The sysfs interface is shown in Table 9.30, “Real-Time Clock sysfs Interface”.

Table 9.30. Real-Time Clock sysfs Interface

sysfs File	Description
<code>since_epoch</code>	Reading this file returns the current number of seconds since the UNIX epoch.
<code>date</code>	Reading this file returns the current date.
<code>time</code>	Reading this file returns the current time.
<code>wakealarm</code>	The alarm interrupt time can be specified by writing either the number of seconds since the UNIX epoch or "+" followed by the number of seconds from the current time. For details refer to Section 9.12.2, “Alarm Interrupts”.

Kernel configuration options related to the real-time clock functionality are shown in `xref linkend="table-realtime-clock-config"/>`[6].

Table 9.31. Real-Time Clock Configuration

Kernel Configuration Options	Default	Description
RTC_CLASS	y	Enables the RTC class
RTC_HCTOSYS	y	Set system time to match the real-time clock value at boot
RTC_HCTOSYS_DEVICE	rtc0	Specify device used when setting system clock at boot
RTC_INTF_SYSFS	y	Enables the sysfs interface
RTC_INTF_PROC	y	Enables the proc interface

[4]N is a numeric character starting from 0.

[5]For example, connecting a Armadillo-400 Series LCD Expansion Board and a WLAN Option Module (AWL12 Compatible and AWL13 Compatible) to Armadillo-440 will result in two real-time clocks being connected. `/dev/rtc0` will correspond to the real-time clock on the WLAN Option Module connected to I2C2 and `/dev/rtc1` will correspond to the real-time clock on the Armadillo-400 Series LCD Expansion Board connected to I2C3.

[6]As real-time clocks are connected to a I2C bus they are also dependent on the I2C configuration. For information on I2C configuration, please refer to Section 9.14, “I2C”.

Kernel Configuration Options	Default	Description
RTC_INTF_DEV	y	Enables the device file interface
RTC_DRV_S35390A	y	Enables the S-35390A driver
RTC_DRV_S353XXA	n	Enables the S-353xxA driver ^[a]

^[a]RTC_DRV_S353XXA was enabled by default instead of RTC_DRV_S35390A up until linux-2.6.26-at9. RTC_DRV_S35390A has been enabled by default from linux-2.6.26-at10 in order to contain the phenomenon that occurs with A400-LCD-Erratum #1. For details on this erratum please refer to the "Armadillo-400 Series Revision Information".

9.12.1. Enabled Real-Time Clock Selection

From linux-2.6.26-at15 and later it is possible to select which real-time clocks to enable from the Linux kernel. Kernel configuration options related to real-time clock selection are shown in Table 9.32, "Real-Time Clock Selection Configuration".

Table 9.32. Real-Time Clock Selection Configuration

Kernel Configuration Options	Default		Description
	Armadillo-420/440	Armadillo-460	
ARMADILLO400_I2C2_CON14_S35390A	y	n	Use S-35390A connected to I2C2 (CON14) as real-time clock ^[a]
ARMADILLO400_I2C3_CON11_S35390A	y	n	Use S-35390A connected to I2C3 (CON11) as real-time clock ^[a]
ARMADILLO460_RTC	n	y	Use RTC on Armadillo-460 board as real-time clock

^[a]Will operate in the same way as linux-2.6.26-at14 and before when this configuration option is enabled for Armadillo-420/440.

9.12.2. Alarm Interrupts

Real-time clock alarm interrupt functionality can be used from the Linux kernel from linux-2.6.26-at13. As alarm interrupts can be made to occur when the CPU is in a sleep state, it is possible to use them to have the CPU return to the executing state from a sleep state. For information on sleep functionality, please refer to Section 9.20, "Power Management". Alarm interrupt times can be specified at one minute intervals up to one week into the future. Seconds are ignored with the alarm occurring at 00 seconds on the specified time (minute).


Table 9.33. Alarm Interrupt Types

Name	Signal Name	Description
Alarm Interrupt 1	RTC_INT1	Mainly used as a basis for sleep wakeup
Alarm Interrupt 2	RTC_INT2	Used to return from a PMIC power off state ^[a] . Nothing will occur if Alarm Interrupt 2 occurs during a PMIC power on state.

^[a]Please refer to Section 9.20.2, "PMIC Power Off Function" for details.

The alarm interrupt functionality can be used with Armadillo-460 and the WLAN Option Module (AWL12 Compatible and AWL13 Compatible)^[7].

On Armadillo-460, the RTC_INT1 signal is connected to the EB0 (GPIO2_12) pin on i.MX257 and the RTC_INT2 signal is connected to the ONOFF signal of the PMIC. After setting the alarm interrupt time on the real-time clock and enabling the alarm interrupt, at the specified time both the RTC_INT1 and RTC_INT2 signals^[8] will change from high to low.



When using RTC_INT2 on Armadillo-460, please connect an external battery to a RTC external backup connector (CON13 or CON20). As power is not supplied to the condenser connected to the real-time clock while the PMIC power is off, if the

^[7]Alarm Interrupt 1 can also be used with the vArmadillo-400 Series RTC Option Module if PD1 is connected to the CON9_2 pin.

^[8]The alarm interrupt times of RTC_INT1 and RTC_INT2 cannot be set individually.

RTC backup time is exceeded the RTC_INT2 will not occur and the board will not return to the executing state.

With the WLAN Option Module (AWL12 Compatible and AWL13 Compatible), the RTC_INT1 signal is connected to the CON9_2 pin. After setting the alarm interrupt time on the real-time clock and enabling the alarm interrupt, at the specified time the RTC_INT1 signal will change from high to low.

The alarm interrupt functionality is disabled in the default Armadillo-420/440 Linux kernel. Therefore the Linux kernel configuration must be changed in order to use the alarm interrupt functionality. Please enable the configuration option shown in Table 9.34, “Real-Time Clock Alarm Function Configuration”.

Table 9.34. Real-Time Clock Alarm Function Configuration

Kernel Configuration Options	Default	Description
ARMADILLO400_RTC_ALM_INT_CON9_2	n	Use CON9_2 for alarm interrupt input
RTC_ALM_INT_WAKE_SRC_SELECT	y	Specify alarm interrupt input as wakeup basis

The alarm interrupt functionality can be used with the device files by using the ioctl system call. Please refer to the sample program described in linux-2.6.26-at/Documents/rtc.txt.

Also, it is possible to use the alarm interrupt functionality by reading and writing to the wakealarm sysfs file. The alarm interrupt time can be set by either writing the seconds since UNIX epoch or "+" followed by the number of seconds from the current time to the wakealarm file. In order to change the alarm interrupt time, the alarm interrupt must be first canceled by writing a past time (or +0) to the file. When an alarm interrupt has been set, reading the file returns the alarm interrupt time. A configuration example using the sysfs file is shown in Figure 9.2, “Alarm Interrupt Time Setting Example”.

```
[armadillo ~]# cat /proc/interrupts | grep rtc0 ❶
142:          0      MXC_GPIO  rtc0
[armadillo ~]# cat /sys/class/rtc/rtc0/since_epoch ❷
1291904885
[armadillo ~]# echo +60 > /sys/class/rtc/rtc0/wakealarm ❸
[armadillo ~]# cat /sys/class/rtc/rtc0/wakealarm ❹
1291904940
:
:
[armadillo ~]# cat /sys/class/rtc/rtc0/since_epoch
1291904945
[armadillo ~]# cat /proc/interrupts | grep rtc0 ❺
142:          1      MXC_GPIO  rtc0
```

Figure 9.2. Alarm Interrupt Time Setting Example

- ❶ The number of interrupts occurred can be checked with /proc/interrupts. Here, the interrupt has not occurred once.
- ❷ The current number of seconds since the UNIX epoch can be checked with since_epoch.
- ❸ +60 is written to wakealarm to set the alarm interrupt time to 60 seconds in the future. Note that as seconds (past the minute) are ignored it is likely that the alarm will not occur exactly 60 seconds later.
- ❹ Reading wakealarm shows that the alarm interrupt time has been set to 55 seconds in the future.
- ❺ Checking the interrupt count after the alarm interrupt time shows that it has increased by one meaning that the interrupt has occurred.

9.13. Watchdog Timer

The i.MX25 processor on the Armadillo-400 Series boards includes an internal watchdog timer.

The default bootloader on the Armadillo-400 Series enables the internal watchdog timer straight after the board is powered on. It is set to time out after 10 seconds by default.

The watchdog timer is kicked automatically by the Linux kernel.

If for some reason the Linux kernel freezes and as a result is unable to kick the watchdog timer and a time-out occurs, the system will reset.

9.14. I2C

The i.MX25 processor includes three I2C controllers: I2C1 to I2C3. On the Armadillo-400 Series, by default I2C1 is used as a board internal bus, I2C2 is assigned to CON14 and I2C3 to CON11.

The I2C bus driver has the following functionality.

- I2C master mode
- 400 kbps max

In order to use a slave device connected to an I2C bus, a chip driver for the appropriate device must be enabled. Also, if the chip driver is written in "new style"^[9], struct `i2c_board_info` must be configured appropriately. For the Armadillo-400 Series, please add the details to the `armadillo400_i2cN_board_info` array in `linux-2.6.26-at/arch/arm/mach-mx25/armadillo400.c` (where N corresponds to the bus number).

The transmission speed of each I2C bus is set to 40kbps by default. The transmission speed is set in `linux-2.6.26-at/arch/arm/mach-mx25/armadillo400.c` at the place shown below. The transmission speed of each bus is set with `i2c_clk`^[10].

```
static struct mxc_i2c_platform_data armadillo400_i2c1_data = {
    .i2c_clk = 400000,
};

static struct mxc_i2c_platform_data armadillo400_i2c2_data = {
    .i2c_clk = 400000,
};

static struct mxc_i2c_platform_data armadillo400_i2c3_data = {
    .i2c_clk = 400000,
};
```

Figure 9.3. I2C Transmission Speed Configuration

Kernel configuration options related to the I2C functionality are shown in Table 9.35, "I2C Configuration".

Table 9.35. I2C Configuration

Kernel Configuration Options	Default	Description
I2C	y	Enables Linux kernel I2C support
I2C_CHARDEV	y	Enables I2C device file interface support

^[9]Refer to `linux-2.6.26-at/Documentation/i2c/writing-clients`.

^[10]The clocks used for the internal modules in i.MX25 are generated from a common reference clock. Because of this the actual transmission speed may differ to that specified with `i2c_clk`. For example, if `400000000(400kbps)` is specified for `i2c_clk`, the actual transmission speed will be 375kbps.

Kernel Configuration Options	Default	Description
I2C_MXC	y	Enables the i.MX I2C driver
ARMADILLO400_I2C2_CON14	y	Enables I2C2 on CON14 Uses CON14_3 for I2C2_SCL and CON14_4 for I2C2_SDA
ARMADILLO400_I2C3_CON11	y	Enables I2C3 on CON11 Uses CON11_48 for I2C3_SCL and CON11_49 for I2C3_SDA

9.15. SPI

The i.MX25 processor includes three SPI controllers: CSPI1 to CSPI3. On the Armadillo-400 Series, CSPI1 and CSPI3 can be assigned to CON9 in the kernel configuration.

The SPI master driver has the following functionality.

- SPI master mode
- Multiple slave selects
- Maximum transmission speed: 16Mbps approx.

The SPI master driver is not enabled by default. In order to use a slave device connected to a SPI bus, both the SPI master driver and a driver for the slave device must be enabled. Also, a struct spi_board_info must be configured appropriately. On the Armadillo-400 Series, please add the details to the armadillo400_spiN_board_info array in linux-2.6.26-at/arch/arm/mach-mx25/armadillo400.c (where N corresponds to the bus number). In contrast to I2C, the SPI bus transmission speed can be configured separately for each slave device.

Kernel configuration options related to the SPI functionality are shown in Table 9.36, “SPI Configuration”.

Table 9.36. SPI Configuration

Kernel Configuration Options	Default	Description
SPI	n	Enables Linux kernel SPI support
SPI_SPIDEV	n	Enables SPI device file interface support
SPI_MXC	n	Enables the i.MX SPI master driver
ARMADILLO400_SPI1_CON9	n	Enables SPI1 on CON9 Uses CON9_3 for CSPI1_MOSI, CON9_5 for CSPI1_MISO, CON9_13 for CSPI1_SCLK and CON9_26 for CSPI1_RDY
ARMADILLO400_SPI1_SS0_CON9_25	n	Uses CON9_25 for SPI1_SS0 Depends on ARMADILLO400_SPI1_CON9
ARMADILLO400_SPI1_SS1_CON9_11	n	Uses CON9_11 for SPI1_SS1 Depends on ARMADILLO400_SPI1_CON9
ARMADILLO400_SPI3_CON9	n	Enables SPI3 on CON9 Uses CON9_4 for CSPI3_MOSI, CON9_6 for CSPI3_MISO, CON9_12 for CSPI3_SCLK and CON9_14 for CSPI3_RDY
ARMADILLO400_SPI3_SS0_CON9_16	n	Uses CON9_16 for SPI3_SS0 Depends on ARMADILLO400_SPI3_CON9
ARMADILLO400_SPI3_SS1_CON9_18	n	Uses CON9_18 for SPI3_SS1 Depends on ARMADILLO400_SPI3_CON9
ARMADILLO400_SPI3_SS2_CON9_15	n	Uses CON9_15 for SPI3_SS2 Depends on ARMADILLO400_SPI3_CON9
ARMADILLO400_SPI3_SS2_CON9_17	n	Uses CON9_17 for SPI3_SS3 Depends on ARMADILLO400_SPI3_CON9

9.16. One Wire

CON9_2 and CON9_26 can be used as one wire masters on the Armadillo-400 Series. The one wire controller included in the i.MX25 processor provides the master functionality for CON9_2, and the GPIO one wire driver provides the same for CON9_26.

The one wire master drivers are not enabled by default. In order to use a slave device connected to a one wire bus, both a one wire master driver and a driver for the slave device must be enabled.

Kernel configuration options related to the one wire functionality are shown in Table 9.37, “One Wire Configuration”.

Table 9.37. One Wire Configuration

Kernel Configuration Options	Default	Description
W1	n	Enables Linux kernel one wire support
W1_MASTER_MXC	n	Enables the i.MX25 internal one wire master controller driver Please enable when using CON9_2
W1_MASTER_GPIO	n	Enables GPIO one wire master driver Please enable when using CON9_26
ARMADILLO400_W1_CON9_2	n	Use CON9_2 as one wire
ARMADILLO400_W1_CON9_26	n	Use CON9_26 as one wire

9.17. PWM

The i.MX25 processor includes four PWM modules: PWM1 to PWM4. On the Armadillo-400 Series, by default PWM1 is used for the LED backlight (CON11_12). PWN2 can be assigned to CON9_25 and PWM4 to CON14_3 by altering the kernel configuration.

With the i.MX25 PWM driver, configuration can be changed by writing an appropriate value to the files under /sys/class/mxc_pwm/(PWM_NAME). Files used for configuration are shown in Table 9.38, “PWM sysfs”.

Table 9.38. PWM sysfs

File Name	Description
period_ns	PWN period set in nsec Acceptable range is from 17 to 2,147,483,647 (from 20usec to 2sec approx.)
duty_ns	PWM on time (off time when invert = 1) set in nsec Acceptable range is 0 < duty_ns < period_ns
invert	PWM output reversed when set to 1
enable	PWM output enabled when set to 1 Output stopped when set to 0 period_ns, duty_ns and invert can be changed even while PWM is enabled

Kernel configuration options related to the PWM functionality are shown in Table 9.39, “PWM Configuration”.

Table 9.39. PWM Configuration

Kernel Configuration Options	Default	Description
MXC_PWM	y	Enable the i.MX25 PWM driver
MXC_PWM_CLASS	y	Enable PWM configuration via sysfs
ARMADILLO400_PWM2_CON9_25	n	Uses CON9_25 as PWM2 The PWM_NAME is CON9_25
ARMADILLO400_PWM4_CON14_3	n	Uses CON14_3 as PWM4 The PWM_NAME is CON14_3

9.18. CAN

The i.MX25 processor includes two CAN controllers (FlexCAN): CAN1 and CAN2. On the Armadillo-400 Series, CAN2 can be assigned to CON14 by altering the kernel configuration. The CAN driver is not enabled by default.

The CAN functionality is provided by the SocketCAN framework. For information on SocketCAN, please refer to linux-2.6.26-at/Documentation/networking/can.txt.

The CAN driver has the following functionality.

- Base and Extended frame format support
- 1Mbps max

Configuration can be changed by writing an appropriate value to the files under /sys/devices/platform/FlexCAN.1/. Files used for configuration are shown in Table 9.40, “CAN sysfs”^[11].

Table 9.40. CAN sysfs

File Name	Description	Default Value	Use Conditions
br_clksrc	Specify clock source When bus is specified, 66.5MHz is used for clock source When osc is specified, 24MHz is used for clock source	bus	A
br_presdiv	Specify clock source prescaler divider 1 to 8 can be specified	7	A
br_propseg	Set propagation segment value 1 to 8 can be specified	5	A
br_pseg1	Set phase buffer segment 1 value 1 to 8 can be specified	5	A
br_pseg2	Set phase buffer segment 2 value 1 to 8 can be specified	8	A
br_rjw	Set resynchronization jump width 1 to 4 can be specified	3	A
bitrate	Displays transmission speed (bps) Cannot be written to	500000	None
std_msg	Set whether or not to support standard format 1 for support, 0 for no support	1	A
ext_msg	Set whether or not to support extended format 1 for support, 0 for no support	1	A
maxmb	Set max message buffer number 2 to 64 can be specified	64	A
rx_maxmb	Set receive message buffer size Send message buffer size is: maxmb - rx_maxmb 1 to maxmb - 1 can be specified	32	A
state	Display current status Displayed with format: "interface status::error status" Interface status is either "Start" (up) or "Stop" (down) Error status is one of following: "normal", "error passive", "bus off"	Stop::normal	None
boff_rec	Set whether or not to recover from bus off automatically 0 to recover automatically, 1 to not recover	1	A
listen	Set whether or not to enable listen mode (receive only) 1 to enable listen mode, 0 to disable	0	A
loopback	Set whether or not to enable loopback mode 1 to enable loopback mode, 0 to disable	0	A
smp	Set sampling behavior 0 for determining received bit value from 1 sample 1 for determining received bit value by majority count from 3 samples	1	A
srx_dis	Set whether or not to receive sent frames 0 to receive sent frames, 1 to not receive	1	A
set_resframe	Set data frame to reply with after receiving remote frame Set with format: "ID#DATA" ID specified with 3 digit hex (standard format) or 8 digit hex value Data specified as 0 to 8 groups of 2 digit hex values Data can be separated by "."	None	B

^[11]Please do not use any file not described here as they are only provided for backwards compatibility.

File Name	Description	Default Value	Use Conditions
del_resframe	Delete data frame set with set_resframe Specify ID of data frame to delete with 3 digit hex (standard format) or 8 digit hex value	None	B
show_resframe	Display data frame set with set_resframe Cannot be written to	None	C
wakeup	Set whether or not to enable CAN receive wakeup while suspended 1 to enable wakeup, 0 to disable	0	A
wak_src	Set whether or not to use low-pass filter while suspended 1 to enable filter, 0 to disable	0	A

- Condition A: can be set when the network interface is in the off state (ifconfig canX off).
- Condition B: can be set when the network interface is in the on state (ifconfig canX on).
- Condition C: can be referenced when the network interface is in the on state (ifconfig canX on).

Transmission speed is determined by the following formula.

```
src_clk = 66,500,000 (br_clksrc = bus condition)
src_clk = 24,000,000 (br_clksrc = osc condition)
Transmission Speed[bps] = src_clk / br_presdiv / (1 + br_propseg + br_pseg1 + br_pseg2)
```

Figure 9.4. CAN Transmission Speed Calculation


Kernel configuration options related to the CAN functionality are shown in Table 9.41, “CAN Configuration”.

Table 9.41. CAN Configuration

Kernel Configuration Options	Default	Description
CAN	n	Enable Linux kernel CAN support
CAN_RAW	n	Enable RAW_CAN protocol support
CAN_BCM	n	Enable CAN_BCM protocol support
CAN_FLEXCAN	n	Enable the i.MX25 FlexCAN driver
ARMADILLO400_CAN2_CON14	n	Uses CON14 for CAN2 Uses CON14_3 for CAN2_TXCAN and CON14_4 for CAN2_RXCAN

9.19. Keypad

The i.MX25 processor includes a keypad controller. On the Armadillo-400 Series, the keypad can be assigned to CON11 by altering the kernel configuration. The keypad driver is not enabled by default.



When the keypad driver is enabled, the event device will be mapped to /dev/input/event0. This will cause the default button and touchscreen event device numbers to change.

Kernel configuration options related to the keypad functionality are shown in Table 9.42, “Keypad Configuration”.

Table 9.42. Keypad Configuration

Kernel Configuration Options	Default	Description
INPUT	y	Enable Linux kernel input layer support

Kernel Configuration Options	Default	Description
INPUT_EVDEV	y	Enable input layer event device support
INPUT_KEYBOARD	y	Enables Linux kernel keyboard support
KEYBOARD_MXC	n	Enable the i.MX25 keypad driver
ARMADILLO400_KEYPAD_CON11	n	Enable keypad on CON11 Please enable at least one COL and ROW each
ARMADILLO400_KEYPAD_ROW0_CON11_40	n	Uses CON11_40 as ROW0
ARMADILLO400_KEYPAD_ROW1_CON11_41	n	Uses CON11_41 as ROW1
ARMADILLO400_KEYPAD_ROW2_CON11_42	n	Uses CON11_42 as ROW2
ARMADILLO400_KEYPAD_ROW3_CON11_43	n	Uses CON11_43 as ROW3
ARMADILLO400_KEYPAD_COL0_CON11_44	n	Uses CON11_44 as COL0
ARMADILLO400_KEYPAD_COL1_CON11_45	n	Uses CON11_45 as COL1
ARMADILLO400_KEYPAD_COL2_CON11_46	n	Uses CON11_46 as COL2
ARMADILLO400_KEYPAD_COL3_CON11_47	n	Uses CON11_47 as COL3
ARMADILLO400_KEYPAD_ROW4_CON11_48	n	Uses CON11_48 as ROW4
ARMADILLO400_KEYPAD_ROW5_CON11_49	n	Uses CON11_49 as ROW5

To set what ROW and COL range is used for the keypad, specify the appropriate values in the `armadillo440_keypad_data` variable in `linux-2.6.26-at/arch/arm/mach-mx25/armadillo400.c`. Specify the button to event key mappings in the `armadillo440_keymapping` variable.

9.20. Power Management

9.20.1. Sleep Function

The Armadillo-400 Series support the Linux power management sleep functionality. In the sleep state, the execution of applications is paused and the kernel enters the suspend state. Power consumption is kept at a minimum during the sleep state as the operation of external devices is halted. When returning to the normal execution state from the sleep state, the kernel's resume logic is run and applications are returned to a running state.

Sleep mode can be entered by writing either "standby" or "mem" to the `/sys/power/state` file. The system will return to the normal execution state from sleep mode when an interrupt from a wakeup basis occurs.

The differences between each sleep state are shown in Table 9.43, "Sleep States". The suspend-to-RAM sleep state provides for a greater reduction in power consumption compared to the power-on suspend state.

Table 9.43. Sleep States

Sleep States	Character String Written to state File	i.MX25 Power Mode	Wakeup Basis
power-on suspend	standby	Doze Mode	Serial input, touchscreen input, button input, alarm interrupt input, CAN input and expansion bus interrupt input ^[a]
suspend-to-RAM	mem	Stop Mode	Button input, alarm interrupt input and CAN input

^[a]Expansion bus interrupt input is only support on Armadillo-460.

For devices that can be used as a basis for wakeups, whether or not they do trigger a wakeup can be specified with their `power/wakeup` sysfs entry. Write "enabled" to the `power/wakeup` file and the device will function as a basis for wakeups, and write "disabled" and they will not. A list of the device `power/wakeup` files is shown in Table 9.44, "Wakeup Basis Designation". Their default values can be changed in the kernel configuration.

Table 9.44. Wakeup Basis Designation

Device	sysfs File	Initial State
Serial Interface 1	<code>/sys/devices/platform/mxcintuart.1/tty/ttymxc1/power/wakeup</code>	enabled
Serial Interface 2	<code>/sys/devices/platform/mxcintuart.2/tty/ttymxc2/power/wakeup</code>	disabled
Serial Interface 3	<code>/sys/devices/platform/mxcintuart.4/tty/ttymxc4/power/wakeup</code>	disabled

Device	sysfs File	Initial State
Serial Interface 4	/sys/devices/platform/mxcintuart.3/tty/ttymxc3/power/wakeup	disabled
Touchscreen	/sys/devices/platform/imx_adc_ts.0/power/wakeup ^[a]	enabled
Buttons	/sys/devices/platform/gpio-keys.0/power/wakeup	enabled
Keypad	/sys/devices/platform/mxc_keypad.0/power/wakeup	enabled
FlexCAN	/sys/devices/platform/FlexCAN.1/wakeup	disabled
Real-time Clock	/sys/devices/platform/i2c-adapter/i2c-1/1-0030/rtc/rtc0/power/wakeup	enabled

^[a]/sys/devices/platform/imx_adc.0/power/wakeup was used until linux-2.6.26-at13, but please use imx_adc_ts for linux-2.6.26-at14 and later.

Table 9.45. Wakeup Basis Default Value Configuration

Kernel Configuration Options	Default		Description
	Armadillo-420/440	Armadillo-460	
ARMADILLO400_UART2_WAKE_SRC_SELECT	y	y	Enables wakeups from UART2 (Serial Interface 1) by default
ARMADILLO400_UART3_WAKE_SRC_SELECT	n	y	Enables wakeups from UART3 (Serial Interface 2) by default
ARMADILLO400_UART4_WAKE_SRC_SELECT	n	y	Enables wakeups from UART4 (Serial Interface 4) by default
ARMADILLO400_UART5_WAKE_SRC_SELECT	n	n	Enables wakeups from UART5 (Serial Interface 3) by default
ARMADILLO400_TOUCHSCREEN_WAKE_SRC_SELECT	y	y	Enables wakeups from the touchscreen by default
ARMADILLO400_GPIO_KEYS_WAKE_SRC_SELECT	y	y	Enables wakeups from buttons by default
ARMADILLO400_RTC_ALM_INT_WAKE_SRC_SELECT ^[a]	n	n	Enables wakeups from alarm interrupt 1 by default
ARMADILLO460_RTC_ALM_INT_WAKE_SRC_SELECT	n	y	Enables wakeups from alarm interrupt 1 by default on Armadillo-460

^[a]Selectable only when CON9_2 alarm interrupt function is enabled.

9.20.1.1. Power Supply to External Devices

Five types of power lines generated by the PMIC (Power Management IC) are used on the Armadillo-400 Series: +5V, +3.3V_CPU, +3.3V_IO, +1.45V and +1.8V^[12]. Of these, the output of +5V and +3.3V_IO can be turned on and off dynamically.

Power is managed with the regulator class in the Armadillo-400 Series Linux kernel, with each PMIC output being assigned as one regulator. The relationship between the power lines and the regulators is shown in Table 9.46, “Power Line and Regulator Relationships”.

Table 9.46. Power Line and Regulator Relationships

Power Lines	Regulator Name	Fixed/Variable
+5V	REG1	Variable
+3.3V_CPU	REG2	Fixed
+1.45V	REG3	Fixed
+1.8V	REG4	Fixed
+3.3V_IO	REG5	Variable

Device drivers of devices that require a power supply can signal that use by obtaining a regulator. As part of the suspend logic when moving to the sleep state, the output of variable regulators (REG1 and REG5) that are not used by any device is turned off. Devices which use regulators are shown in Table 9.47, “Regulators Used By Devices”.

^[12]Refer to "Power Circuit Makeup" in the "Armadillo-400 Series Hardware Manual".

Table 9.47. Regulators Used By Devices

Device	Regulator Name
Serial Interface 2 (UART3)	REG5
Serial Interface 3 (UART5)	REG5
Touchscreen	REG5
MMC/SD/SDIO Host (eSDHC1) ^[a]	REG2
MMC/SD/SDIO Host (eSDHC2)	REG5
USB ^[b]	REG1

^[a]Armadillo-420/440 only

^[b]Armadillo-420/440 only

Of these, the power supply to microSD/SD and USB is handled specially. Power supply to microSD(eSDHC1) via a power switch from +3.3V_CPU on Armadillo-420/440 and +3.3V_EXT on Armadillo-460. While the +3.3V_CPU and +3.3V_EXT themselves cannot be turned on and off, the power supplied to the microSD/SD can be by controlling the GPIO connected to the power switch. This functionality is used in the suspend logic to halt the power supply to the microSD/SD card.

Either VIN (Power Input) or the +5V output from the PMIC can be selected for the power supply to USB on Armadillo-420/440^[13]. VIN is used by default. To use the PMIC +5V output, in `linux-2.6.26-at/arch/arm/mach-mx25/board-armadillo400.h` define `USB_PWRSRC` as `USB_PWRSRC_5V` and build the kernel. Whichever power source is used for USB, the power supply is turned off in the suspend logic.

The behavior of +3.3V_IO (REG5) during the sleep state changes depending on the Armadillo-400 Series model and whether or not the wakeup of the devices used by that model have been enabled or not. On the Armadillo-420 basic model, Serial Interface 2 and 3 use REG5. However, as the wakeup on Serial Interface 2 and 3 is disabled by default, +3.3V_IO is turned off during the sleep state. If the wakeup is enabled for either serial interface +3.3V_IO will continue to be output during the sleep state.

On the Armadillo-420 WLAN Model (AWL12 Compatible) and Armadillo-420 WLAN Model (AWL13 Compatible), Serial Interface 2 and 3 and MMC/SD/SDIO Host (eSDHC2) use REG5. As the wakeup is disabled by default for all of these devices, +3.3V_IO is turned off during the sleep state. In the same way as the Armadillo-420 Basic Model, if the wakeup is enabled for either serial interface +3.3V_IO will continue to be output during the sleep state.

On the Armadillo-440 LCD Model, Serial Interface 2 and 3 and the touchscreen use REG5. As the touchscreen wakeup is enabled by default, +3.3V_IO continues to be output during the sleep state^[14]. If the touchscreen wakeup is disabled, +3.3V_IO will be turned off during the sleep state.

On the Armadillo-460 Basic Model, Serial Interface 2 and 3 and the touchscreen^[15] use REG5. As wakeup is enabled for Serial Interface 2 and 3 and the touchscreen by default, +3.3V_IO continues to be output during the sleep state^[16]. If wakeup for Serial Interface 2 and 3 and the touchscreen is disabled, +3.3V_IO will be turned off during the sleep state.

9.20.1.2. Treatment of External Devices During Sleep

As stated in the preceding section, the power supply to USB and microSD/SD is turned off in the suspend logic during the move to the sleep state on the Armadillo-400 Series.

Because of this, USB devices must be put into a state where they can be safely disconnected before the system is put into the sleep state. That is, USB memory must be unmounted first. As the devices will be detected again when the system resumes, USB devices can be disconnected and reconnected while the system is in the sleep state.

As opposed to this, microSD/SD cards can be left mounted when the system is put into the sleep state. In order for this to be possible, the MMC/SD/SDIO host driver does not probe for cards at resume time, and instead assumes the same card is still inserted. Because of this, microSD/SD cards cannot be removed or inserted when the system is in the sleep state.

^[13]For Armadillo-460, the USB power supply is fixed with VIN (Power Input).

^[14]As the touchscreen did not use REG5 in linux-2.6.26-at13 and previous kernels, +3.3V_IO was turned off during the sleep state.

^[15]Usable when the LCD Expansion Board is connected to Armadillo-400 Series

^[16]

For Ethernet devices, at resume time the same process is followed as when a cable is reconnected. Therefore, Auto-negotiation will take place at resume time if it is enabled.

9.20.2. PMIC Power Off Function

In Linux kernels linux-2.6.26-at15 and later, the PMIC (Power Management IC) power supply can be moved to the OFF state by writing off to the `/sys/power/supply` file^[17]. In this state, as all power supply from the PMIC (including to the CPU) is halted, power consumption is even lower than when in the sleep state introduced in Section 9.20.1, “Sleep Function”. On Armadillo-460, Alarm Interrupt 2 can be used to move from the power OFF state to the ON state^[18]. For information on using Alarm Interrupt 2, please refer to Section 9.12.2, “Alarm Interrupts”.



After cutting the power supply to Armadillo-460 while the PMIC is in the power OFF state, the PMIC may not return from the power OFF state when reconnecting the power supply. This occurs when the condenser connected to the PMIC power supply has not discharged and has maintained the PMIC in the power OFF state. In order to ensure the PMIC returns from the power OFF state, short the CON13_4 pin to GND or use Alarm Interrupt 2.



Even when the PMIC is in the power OFF state, power will still be supplied to the LCD Interface and the PC/104 Expansion Connector on Armadillo-460 as their power supply does not go through the PMIC, and also to the microSD slot on Armadillo-420/440 and the USB interface on Armadillo-460 when their power switch is not turned off as their power supply also does not go through the PMIC. Power consumption may increase if an external device is connected to these interfaces.

^[17]The PMIC power supply can also be moved to the OFF state by shorting the CON13_4 pin to GND for more than two seconds.

^[18]The PMIC power supply can also be moved to the ON state by shorting the CON13_4 pin to GND.

Chapter 10. Armadillo-460 Expansion Bus

This chapter describes the expansion bus on Armadillo-460 in the Armadillo-400 Series.

This expansion bus employs a PC/104 compliant bus layout and can be used in either the "PC/104 Expansion Bus Compatibility Mode" or the "Direct CPU Bus Mode". With the "PC/104 Expansion Bus Compatibility Mode" it is possible to easily expand functionality by connecting a third party PC/104 expansion board^[1]. With the "Direct CPU Bus Mode", the expansion bus signal lines are connected directly to the CPU making high speed access possible.

There are two modes that can be selected for the "Direct CPU Bus Mode": "Synchronous Mode" where the control signals are synchronized to SYSCLK (66MHz), and "Asynchronous Mode" where it is possible to configure any access timing.

The Expansion Bus interrupt controller and configuration registers are implemented in the CPLD. For detailed specifications on the CPLD, please refer to the "Armadillo-400 Series Hardware Manual".

10.1. Kernel Configuration Options

The following describes kernel configuration options related to the Armadillo-460 Expansion Bus.

10.1.1. Expansion Bus Operating Modes

Kernel configuration options related to the operating mode of the Armadillo-460 Expansion Bus are shown in Table 10.1, "Armadillo-460 Expansion Bus Operating Mode Configuration".

Table 10.1. Armadillo-460 Expansion Bus Operating Mode Configuration

Kernel Configuration Options	Default	Description
ARMADILLO460_EXT_BUS_PC104_MODE	y	Sets the Expansion Bus operating mode to the PC/104 Expansion Bus Compatibility Mode
ARMADILLO460_EXT_BUS_DIRECT_CPU_SYNC_MODE	n	Sets the Expansion Bus operating mode to the Direct CPU Bus Mode (Synchronous)
ARMADILLO460_EXT_BUS_DIRECT_CPU_ASYNC_MODE	n	Sets the Expansion Bus operating mode to the Direct CPU Bus Mode (Asynchronous)

10.1.2. Direct CPU Bus Mode (Synchronous) Only

Kernel configuration options unique to the Direct CPU Bus Mode (Synchronous)^[2] are shown in Table 10.2, "Direct CPU Bus Mode (Synchronous) Configuration".

Table 10.2. Direct CPU Bus Mode (Synchronous) Configuration

Kernel Configuration Options	Default	Description
ARMADILLO460_EXT_BUS_DIRECT_CPU_INVERTED_SYSCLK	n	Inverts ^[a] the SYSCLK phase output to the connector
ARMADILLO460_EXT_BUS_DIRECT_CPU_CS3_8BIT	y	Sets the data bus width of Chip Select 3 to 8bits
ARMADILLO460_EXT_BUS_DIRECT_CPU_CS3_16BIT	n	Sets the data bus width of Chip Select 3 to 16bits

^[a]The signal lines latch on the rising edge of SYSCLK by default.

^[1]Please check the Armadillo Site Armadillo-460 Product Page for regularly updated information on tested PC/104 boards and HOWTOs describing how to use the PC/104 boards.

^[2]Can only be set when Direct CPU Bus Mode (Synchronous) has been selected in Section 10.1.1, "Expansion Bus Operating Modes"

10.1.3. Direct CPU Bus Mode (Asynchronous) Only

Kernel configuration options unique to the Direct CPU Bus Mode (Asynchronous)^[3] are shown in Table 10.3, “Direct CPU Bus Mode (Asynchronous) Configuration”. Bus timing can be configured by specifying the values of the Wireless External Interface Module (WEIM) registers in i.MX257^[4].

Table 10.3. Direct CPU Bus Mode (Asynchronous) Configuration

Kernel Configuration Options	Default	Description
ARMADILLO460_EXT_BUS_DIRECT_CPU_CS3_CSCRU	0x00000000	Sets the value of the WEIM CSCR3U register
ARMADILLO460_EXT_BUS_DIRECT_CPU_CS3_CSCRL	0x00000000	Sets the value of the WEIM CSCR3L register
ARMADILLO460_EXT_BUS_DIRECT_CPU_CS3_CSCRA	0x00000000	Sets the value of the WEIM CSCR3A register
ARMADILLO460_EXT_BUS_DIRECT_CPU_CS3_AUS	y	Sets the AUS3 bit of the WEIM WCR register to 1.
ARMADILLO460_EXT_BUS_DIRECT_CPU_CS4_CSCRU	0x00000000	Sets the value of the WEIM CSCR4U register
ARMADILLO460_EXT_BUS_DIRECT_CPU_CS4_CSCRL	0x00000000	Sets the value of the WEIM CSCR4L register
ARMADILLO460_EXT_BUS_DIRECT_CPU_CS4_CSCRA	0x00000000	Sets the value of the WEIM CSCR4A register
ARMADILLO460_EXT_BUS_DIRECT_CPU_CS4_AUS	y	Sets the AUS4 bit of the WEIM WCR register to 1.

10.2. Memory Map

The memory map of the Armadillo-460 Expansion Bus differs depending on the operating mode. The memory map for the PC/104 Expansion Bus Compatibility Mode is shown in Table 10.4, “PC/104 Expansion Bus Compatibility Mode Memory Map”, and the memory map for the Direct CPU Bus Mode is shown in Table 10.5, “Direct CPU Bus Mode Memory Map”.

In Direct CPU Bus Mode, depending on the kernel configuration not all of the CS3/CS4 space shown in Table 10.5, “Direct CPU Bus Mode Memory Map” can be accessed. The accessible space is shown in Table 10.6, “Accessible CS3/CS4 Space In Direct CPU Bus Mode”.

Table 10.4. PC/104 Expansion Bus Compatibility Mode Memory Map

Chip Select	Physical Address	Virtual Address	Access Width	Description
CS1	0xa8000000 0xa800000f	0xe8000000 0xe800000f	8bit	CPLD Registers
	0xa8000010 0xafffffff	0xe8000010 0xefffffff	-	Reserved
CS3	0xb2000000 0xb200ffff	0xf2000000 0xf200ffff	8bit	PC/104 I/O Space
	0xb2010000 0xb2ffffff	0xf2010000 0xf2ffffff	-	Reserved
	0xb3000000 0xb3ffffff	0xf3000000 0xf3ffffff	8bit	PC/104 Memory Space

^[3]Can only be set when Direct CPU Bus Mode (Asynchronous) has been selected in Section 10.1.1, “Expansion Bus Operating Modes”

^[4]For details on the WEIM registers, please refer to the “i.MX25 Multimedia Applications Processor Reference Manual” stored in the /document/datasheet directory on the included DVD.

Chip Select	Physical Address	Virtual Address	Access Width	Description
CS4	0xb400000 0xb400fff	0xf400000 0xf400fff	16bit	PC/104 I/O Space
	0xb401000 0xb4fffff	0xf401000 0xf4fffff	-	Reserved
CS4	0xb500000 0xb57ffff	0xf500000 0xf57ffff	16bit	PC/104 Memory Space
	0xb580000 0xb5fffff	0xf580000 0xf5fffff	-	Reserved

Table 10.5. Direct CPU Bus Mode Memory Map

Chip Select	Physical Address	Virtual Address	Access Width	Description
CS1	0xa800000 0xa80000f	0xe800000 0xe80000f	8bit	CPLD Registers
	0xa800010 0xaffffff	0xe800010 0xefffff	-	Reserved
CS3	0xb200000 0xb3fffff	0xf200000 0xf3fffff	8/16bit	CS3 Space
CS4	0xb400000 0xb5fffff	0xf400000 0xf5fffff	8/16bit	CS4 Space ^[a]

^[a]Cannot be used in synchronous mode.

Table 10.6. Accessible CS3/CS4 Space In Direct CPU Bus Mode

Chip Select	Physical Address	Synchronous Mode ^[a]		Asynchronous Mode ^[b]			
		8bit ^[c]	16bit ^[d]	8bit ^[e]	8bit AUS ^[f]	16bit ^[g]	16bit AUS ^[h]
CS3	0xb200000 0xb27ffff	16MByte	8MByte	16MByte	8MByte	32MByte	8MByte
	0xb280000 0xb2fffff		Reserved		Reserved		Reserved
	0xb300000 0xb37ffff	Reserved	Reserved	Reserved	Reserved		Reserved
	0xb380000 0xb3fffff						

Chip Select	Physical Address	Synchronous Mode ^[a]		Asynchronous Mode ^[b]			
		8bit ^[c]	16bit ^[d]	8bit ^[e]	8bit AUS ^[f]	16bit ^[g]	16bit AUS ^[h]
CS4	0xb4000000 0xb47fffff	Reserved	Reserved	16MByte	8MByte	32MByte	8MByte
	0xb4800000 0xb4ffffff				Reserved		Reserved
	0xb5000000 0xb57fffff			Reserved			
	0xb5800000 0xb5ffffff				Reserved		Reserved

^[a]When Direct CPU Bus Mode (Synchronous) is set in Section 10.1.1, “Expansion Bus Operating Modes”

^[b]When Direct CPU Bus Mode (Asynchronous) is set in Section 10.1.1, “Expansion Bus Operating Modes”

^[c]When data bus width is set to 8bit in Section 10.1.2, “Direct CPU Bus Mode (Synchronous) Only”


^[d]When data bus width is set to 16bit in Section 10.1.2, “Direct CPU Bus Mode (Synchronous) Only”

^[e]When data bus width is set to 8bit and AUS3/AUS4 bits in the WEIM WCR register are not set to 1 in Section 10.1.3, “Direct CPU Bus Mode (Asynchronous) Only”

^[f]When data bus width is set to 8bit and AUS3/AUS4 bits in the WEIM WCR register are set to 1 in Section 10.1.3, “Direct CPU Bus Mode (Asynchronous) Only”

^[g]When data bus width is set to 16bit and AUS3/AUS4 bits in the WEIM WCR register are not set to 1 in Section 10.1.3, “Direct CPU Bus Mode (Asynchronous) Only”

^[h]When data bus width is set to 16bit and AUS3/AUS4 bits in the WEIM WCR register are set to 1 in Section 10.1.3, “Direct CPU Bus Mode (Asynchronous) Only”



Due to the Errata ENGcm11270 limitation, A[23] cannot be used when AUS (Address Unshifted mode) is specified. Because of this, the address space is limited depending on the Direct CPU Bus Mode configuration. For information on this errata, please refer to "ENGcm11270" in "Chip Errata for the i.MX25" from the / document/datasheet/ directory on the included DVD.

10.3. Precautions When Using ISA Drivers

With standard ISA, the I/O space and memory space exist in different spaces even when the addresses appear the same. As memory mapped I/O is used on Armadillo-460, I/O and memory exist in the same address space. Because of this, some of the ISA drivers included in the Linux kernel may not be usable as is.

On Armadillo-460 the memory maps are as shown in Section 10.2, “Memory Map”. As a 8bit accessible space and a 16bit accessible space exist for each chip select space, access must be explicitly separated between the spaces.

On CPUs (SOCs) that support ISA, the ISA interrupts are mapped directly. However, as interrupts specifically for ISA do not exist on Armadillo-460, they must be controlled via the expansion interrupt controller included in the CPLD. Therefore, the ISA interrupt numbers differ to those used with Armadillo-460. The Armadillo-460 interrupt numbers must be used for software control. The Armadillo-460 interrupt numbers are shown in Table 10.7, “Interrupt Signals and Corresponding Armadillo-460 Interrupt Numbers”.

Table 10.7. Interrupt Signals and Corresponding Armadillo-460 Interrupt Numbers

Signal Name	Interrupt Number
IRQ3	195 ^[a]
IRQ4	196 ^[a]
IRQ5	197 ^[a]

Signal Name	Interrupt Number
IRQ6	198 ^[a]
IRQ7	199 ^[a]
IRQ9	201 ^[a]
IRQ10	202 ^[a]
IRQ11	203 ^[a]
IRQ12	204 ^[b]
IRQ14	206 ^[b]
IRQ15	207 ^[b]

^[a]Interrupt level and type can be selected from: LEVEL-HIGH, LEVEL-LOW, RISING-EDGE and FALLING-EDGE.


^[b]Interrupt level and type fixed at LEVEL-HIGH.

10.4. Header File for Armadillo-460 PC/104

As explained in Section 10.3, “Precautions When Using ISA Drivers”, some ISA drivers included in the Linux kernel may not be usable as is on Armadillo-460. In those cases, the Linux kernel driver must be modified. A number of macros have been made available in order to adjust the ISA driver to the PC/104 specifications on Armadillo-460. These macros are stored in the `include/asm-arm/arch-mxc/armadillo460_extbus.h` Linux kernel source file. Use the following statement to include this file.

```
#include <asm/arch/armadillo460_extbus.h>
```

Figure 10.1. Including the PC/104 Header File



Make sure to include this header after `<asm/io.h>`. If it is not included in the correct place, the following error will occur.

```
include/asm/arch/armadillo460_extbus.h:142:2: error: #error
you must include this file after "asm/io.h".
```

10.4.1. I/O Port Access Macros

The macros `outw`, `inw`, `outsw` and `insw` for I/O port access are replaced with versions for PC/104 access on Armadillo-460, making correct access to the 8/16bit I/O spaces possible^[5]. The parameters for each macro are the same as those defined in `<asm/io.h>`.

10.4.2. Interrupt Number Conversion Marcos

These macros can be used to convert the ISA interrupt numbers and the interrupt numbers used with Armadillo-460. For the relationship between the ISA and Armadillo-460 interrupt numbers, please refer to Table 10.7, “Interrupt Signals and Corresponding Armadillo-460 Interrupt Numbers”.

Table 10.8. Interrupt Number Conversion Macro Specifications

Macro	Description	Usage Example
<code>convirq_to_isa(irq)</code>	Converts Armadillo-460 interrupt numbers to ISA interrupt numbers.	<code>convirq_to_isa(195); // converts to 3</code>
<code>convirq_from_isa(irq)</code>	Converts ISA interrupt numbers to Armadillo-460 interrupt numbers.	<code>convirq_from_isa(3); // converts to 195</code>

^[5]As the PC/104 I/O space has a maximum data bus width of 16bit, 32bit access cannot be performed. Therefore please do not use the `outl`, `inl`, `outsl` and `insl` access macros.

Appendix A. Hermit-At Bootloader

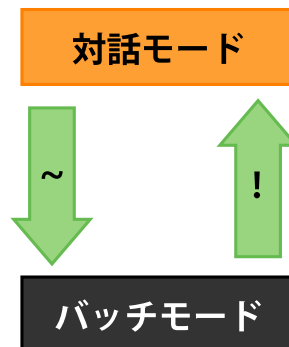
Hermit-At is a functional downloader and bootloader used on Atmark Techno products. The Hermit-At bootloader prompt is displayed when an Armadillo board is booted in maintenance mode. From the prompt it is possible to enter commands for a variety of operations such as updating flash memory and setting Linux kernel parameters. The following gives details on the most well used functions.



Hermit-At Modes

Hermit-AT has two modes: the "interactive mode" which displays the command prompt and operates interactively, and the "batch mode" for communicating with the Hermit-AT downloader. The command prompt and entered characters are not displayed in batch mode, but commands can be executed.

Hermit-AT is always in interactive mode after booting. To move from interactive mode to batch mode enter a tilde "~" and to move from batch mode to interactive mode enter an exclamation mark "!".



Hermit-AT moves to batch mode when communicating with the Hermit-AT downloader. This is because the Hermit-AT downloader sends a tilde in order to establish the communication.

When moving from interactive mode to batch mode and when an entered command succeeds in batch mode the following is displayed.

```
+OK
```

A.1. version

This command displays version information.

```
Syntax: version
```

Figure A.1. version Syntax

A.1.1. version Example

```
hermit> version  
Hermit-At v2.0.0 (armadillo4x0) compiled at 23:03:08, Mar 08 2010
```

Figure A.2. version Example

A.2. info

This command displays board information.

```
Syntax: info
```

Figure A.3. info Syntax

A.2.1. info Example

```
hermit> info  
Board Type: 0x00000440  
Hardware ID: 0x00000300  
  DRAM ID: 0x00000002  
  Jumper: 0x00000001  
  Tact-SW: 0x00000000
```

Figure A.4. info Example

A.3. memmap

This command displays the flash memory and DRAM memory map.

```
Syntax: memmap
```

Figure A.5. memmap Syntax

A.3.1. memmap Example

```
hermit> memmap
0xa0000000:0xa1fffffff FLA all bf:8K bl:4x32K/l,255x128K/l
0xa0000000:0xa001ffff FLA bootloader bf:8K bl:4x32K/l
0xa0020000:0xa021ffff FLA kernel bf:8K bl:16x128K
0xa0220000:0xa1fdffff FLA userland bf:8K bl:238x128K
0xa1fe0000:0xa1fffffff FLA config bf:8K bl:1x128K
0x80000000:0x87fffffff RAM dram-1
```

Figure A.6. memmap Example

A.4. mac

This command displays the MAC address.

```
Syntax: mac
```

Figure A.7. mac Syntax

A.4.1. mac Example

```
hermit> mac
00:11:0c:00:00:00
```

Figure A.8. mac Example

A.5. md5sum

This command calculates and displays the md5sum value of the specified memory region.

```
Syntax: md5sum <start address> <size>
```

Figure A.9. md5sum Syntax

A.5.1. md5sum Example

To calculate and display the md5sum value of the first 1024 bytes of the bootloader region, execute the command shown in Figure A.10, “md5sum Example”.

```
hermit> memmap
0xa0000000:0xa1ffffff FLA all bf:8K bl:4x32K/1,255x128K/1
0xa0000000:0xa001ffff FLA bootloader bf:8K bl:4x32K/1
0xa0020000:0xa021ffff FLA kernel bf:8K bl:16x128K
0xa0220000:0xa1fdffff FLA userland bf:8K bl:238x128K
0xa1fe0000:0xa1ffffff FLA config bf:8K bl:1x128K
0x80000000:0x87ffffff RAM dram-1
hermit> md5sum 0xa0000000 1024
fd44ce938f65726dc59669f537154429
```

Figure A.10. md5sum Example

A.6. erase

This command erases flash memory.

```
Syntax: erase [address]
```

Figure A.11. erase Syntax

A.6.1. erase Example

```
hermit> erase 0xa0fe0000
```

Figure A.12. erase Example

A.7. setenv and clearenv

These commands are used to set Linux kernel parameters. The parameters set with setenv are passed to the kernel at boot time. Executing clearenv clears any configuration. The parameters are saved in flash memory and are therefore maintained even after rebooting.

```
Syntax: setenv [kernel parameter]...
```

```
Explanation: Sets kernel parameters. Shows current configuration if executed without any options specified.
```

```
Syntax: clearenv
```

```
Explanation: Clears all set options.
```

Figure A.13. setenv/clearenv Syntax

A.7.1. setenv/clearenv Example

```

hermit> setenv console=ttymxc1
hermit> setenv
1: console=ttymxc1
hermit> clearenv
hermit> setenv
hermit>
    
```


Figure A.14. setenv and clearenv Example

A.7.2. Linux Kernel Parameters

Example Linux kernel parameters are shown in Table A.1, “Well Used Linux Kernel Parameters”. For information on other options, please refer to `linux-2.6/Documentation/kernel-parameters.txt`.

Table A.1. Well Used Linux Kernel Parameters

Option	Description
console	Specify device to be used for kernel console.
root	Specify root filesystem related settings.
rootdelay	Seconds to wait before attempting to mount root filesystem.
rootwait	Wait until the root filesystem is accessible before attempting to mount it.
noinitrd	Specify what should happen with the initrd data after the kernel has booted.
nfsroot	Specify root filesystem place and NFS options when using NFS.



When `ttymxc1,2,4` is specified for the console option, the serial interface used by Hermit-At will also change to that interface after the next boot.

A.8. setbootdevice

This command is used to specify the boot device holding the Linux kernel. This setting is saved in flash memory and is therefore maintained even after rebooting.

Syntax: `setbootdevice flash`
 Explanation: Extract the kernel image stored in the kernel flash memory region to RAM and boot it

Syntax: `setbootdevice tftp <client IP address> <server IP address> [--kernel=<path>] [--userland=<path>]`
 Explanation: Download kernel and/or userland image from TFTP server, extract to RAM and boot

Syntax: `setbootdevice mmcblk0pN`
 Explanation: Extract kernel image from `/boot/` directory in partition N of MMC/SD card to RAM and boot

Figure A.15. setbootdevice Syntax

A.8.1. setbootdevice Example

To boot the kernel image stored in flash memory, execute the command shown in Figure A.16, “Assigning Flash Memory As Boot Device”.

```
hermit> setbootdevice flash
```

Figure A.16. Assigning Flash Memory As Boot Device

To download and boot a kernel image named `linux.bin.gz` from a TFTP server (IP address: 192.168.10.1), execute the command shown in Figure A.17, “Assigning TFTP Server As Boot Device”.

```
hermit> setbootdevice tftp 192.168.10.10 192.168.10.1 --kernel=linux.bin.gz
```

Figure A.17. Assigning TFTP Server As Boot Device



The client IP address is an IP address that is used temporarily to communicate with the TFTP server. Please choose an IP address that is not already in use on the network.

To boot a kernel image stored in the first partition of a SD/MMC card, execute the command shown in Figure A.18, “Assign SD/MMC Card As Boot Device”.

```
hermit> setbootdevice mmcblk0p1
```

Figure A.18. Assign SD/MMC Card As Boot Device

A.9. frob

This command is used to enter a mode for reading or altering data at a specified address.

Table A.2. frob Command

frob Command	Description
peek [addr]	Read 32bit data from specified address
peek16 [addr]	Read 16bit data from specified address
peek8 [addr]	Read 8bit data from specified address
poke [addr] [value]	Write 32bit data to specified address
poke16 [addr] [value]	Write 16bit data to specified address
poke8 [addr] [value]	Write 8bit data to specified address

A.10. tftpd

This command is used to download an image file from a TFTP server and write it to flash memory.

Syntax: `tftpd <client IP address> <server IP address> <option> [option]...`

Explanation: Sets the local IP address to the specified client address, then downloads an image file from the TFTP server at the specified server IP address and writes it to flash memory.

Figure A.19. tftpd Syntax

Table A.3. tftpd Options

Option	Description
--bootloader=filepath	Specify file to be written to the bootloader region in place of filepath.
--kernel=filepath	Specify file to be written to the kernel region in place of filepath.
--userland=filepath	Specify file to be written to the userland region in place of filepath.
--fake	Download the files, but do not actually write the files to flash memory.

A.10.1. tftpd Example

```

hermit> tftpd 192.168.10.10 192.168.10.1 --kernel=linux.bin.gz


Client: 192.168.10.10
Server: 192.168.10.1
Region(kernel): linux.bin.gz

initializing net-device...OK
Filename : linux.bin.gz
.....
.....
.....
Filesize : 1841551

programing: kernel
#####

completed!!
    
```

Figure A.20. tftpd Example



The client IP address is an IP address that is used temporarily to communicate with the TFTP server. Please choose an IP address that is not already in use on the network.

A.11. tftpboot

This command is used to download an image file from a TFTP server, extract it to RAM and then boot it. As opposed to the tftpd command, the file is not written to flash memory. Also, the settings are not saved like they are when using the setbootdevice command.

Syntax: tftpboot <client IP address> <server IP address> <option> [option]...

Explanation: Sets the local IP address to the specified client address, then downloads the specified image file from the TFTP server at the specified server IP address, extracts it to RAM and boots it.

Figure A.21. tftpboot Syntax

The same options as shown in Table A.3, “tftpd Options” can be used with this command. When --fake is specified, the file is downloaded but not booted.

A.11.1. tftpboot Example

```

hermit> tftpboot 192.168.10.10 192.168.10.1 --kernel=linux.bin.gz

Client: 192.168.10.10
Server: 192.168.10.1
Region(kernel): linux.bin.gz


initializing net-device...OK
Filename : linux.bin.gz
.....
.....
.....
Filesize : 1841551

Uncompressing kernel:net.....done.
Uncompressing ramdisk.....done.
.....
.....
.....
.....
.....done.
Linux version 2.6.26-at6 (2.6.26) (atmark@sv-build) (gcc version 4.3.2 (Debian
4.3.2-1.1) ) #6 PREEMPT Wed Mar 10 19:19:13 JST 2010
:
:

```

Figure A.22. tftpboot Example

- ❶ The kernel and userland images are being extracted to RAM.
- ❷ The kernel is booted and kernel boot log displayed.



The client IP address is an IP address that is used temporarily to communicate with the TFTP server. Please choose an IP address that is not already in use on the network.

A.12. boot

This command boots the Linux kernel image from the boot device specified with the setbootdevice command.

```
Syntax: boot
```

Figure A.23. boot Syntax

A.12.1. boot Example

```

hermit> boot
Uncompressing kernel.....done.
Uncompressing ramdisk.....
.....
.....done.
Doing console=ttymxc1
Linux version 2.6.26-at6 (2.6.26) (atmark@sv-build) (gcc version 4.3.2 (Debian
4.3.2-1.1) ) #6 PREEMPT Wed Mar 10 19:19:13 JST 2010
CPU: ARM926EJ-S [41069264] revision 4 (ARMv5TEJ), cr=00053177
Machine: Armadillo-440
Memory policy: ECC disabled, Data cache writeback
CPU0: D VIVT write-back cache
CPU0: I cache: 16384 bytes, associativity 4, 32 byte lines, 128 sets
CPU0: D cache: 16384 bytes, associativity 4, 32 byte lines, 128 sets
Built 1 zonelists in Zone order, mobility grouping on. Total pages: 32512
Kernel command line: console=ttymxc1
MXC IRQ initialized
:
:

```

Figure A.24. boot Example

- ❶ The kernel and userland images are being extracted to RAM.
- ❷ Kernel parameters set with the setenv command are displayed. All messages up to this point are from Hermit-At.
- ❸ The kernel is booted and kernel boot log displayed.

A.13. Note on Versions

When a bootloader image based on hermit-at v2.0.0 source (loader-armadillo4x0-v2.0.0.bin etc) and a kernel image based on linux-2.6.26-at7 source (linux-a400-1.00.bin.gz etc) are used on Armadillo-440 Rev.C1 or later boards (S/N 100201-2195 or greater), a problem exists where the kernel may not boot. This problem only occurs with the above combination.

On Armadillo-440 Rev.C1 or later boards (S/N 100201-2195 and greater), please use a bootloader image based on hermit-at v2.0.1 or later source (loader-armadillo4x0-v2.0.1.bin or later).^[1]

^[1]On Armadillo-440 Rev.C1 and later boards, loader-armadillo4x0-v2.0.1.bin or a later version is written to flash memory in the default factory state.

Revision History

Revision	Date	Description
1.0.0	03/12/2010	<ul style="list-style-type: none"> Initial Release
1.1.0	04/28/2010	<ul style="list-style-type: none"> Added information on Armadillo-420 to Chapter 1, Preface, Chapter 3, System Overview, Chapter 4, Before Getting Started, Chapter 6, Rewriting Flash Memory, Chapter 7, Building and Chapter 8, Kernel and User-land Placement. Added notes on function change of pins 3,4 on CON14 to Section 3.5, "Basic Specifications of Armadillo-440 LCD Model". Corrected directory name in Figure 7.14, "Hermit-At Source Archive Extraction". Fixed URLs in Table 8.1, "Kernel Image Download URLs", Figure 8.5, "Kernel Image Placement", Table 8.2, "Debian Archive Download URLs", Figure 8.6, "Root Filesystem Creation with Debian Archives", Table 8.3, "Atmark-Dist Image Download URL" and Figure 8.6, "Root Filesystem Creation with Debian Archives". Added note about using board revision Rev.C1 to Section A.13, "Note on Versions".
1.2.0	06/08/2010	<ul style="list-style-type: none"> Made correction to description of CON9 5 functionality in Table 3.3, "Default States of Armadillo-420 Basic Model Expansion Interfaces" and Table 3.9, "Default States of Armadillo-440 LCD Model Expansion Interfaces". Added notes on changing configuration to Section 7.1.4, "Customizing Images". Added notes on drivers that can be configured in the kernel configuration to Chapter 9, Linux Kernel Device Driver Specifications.
1.3.0	08/20/2010	<ul style="list-style-type: none"> Made corrections to explanation of RS232 connections in Section 3.2, "Basic Specifications of Armadillo-420 Basic Model" Made corrections to Figure 9.4, "CAN Transmission Speed Calculation" Added use conditions to Table 9.40, "CAN sysfs" Added UART4 to Table 9.2, "Serial Interface Device Files" Updated explanation in Section 8.2, "Loading from Storage" Corrected inconsistent use of various terms Added explanation on how to use drivers not enabled by default to Chapter 9, Linux Kernel Device Driver Specifications Add notes on how to handle interrupts with GPIO sysfs to Section 9.9.1, "GPIO sysfs"
1.4.0	12/24/2010	<ul style="list-style-type: none"> Added example of altering fstab to Section 8.2.4.2, "Using an Atmark-Dist Image" Added note about reserved regions to Section 2.3, "Software Usage Precautions" Corrected an error related to audio multiplex in Section 9.8, "Audio" Made corrections to Figure 9.1, "GPIO sysfs Interrupt Sample Program" Made updates for Hermit-At Win32 v1.3.0 Corrected inconsistent use of various terms Added parameter erasing to Section 6.6, "Restoring Bootloader to Factory State" Updated Chapter 9, Linux Kernel Device Driver Specifications for linux-2.6.26-at13 Added configuration information for each section in Chapter 9, Linux Kernel Device Driver Specifications Moved precautions to Chapter 2, Precautions Added explanation of jumper settings

		<ul style="list-style-type: none"> • Corrected inconsistent use of product names
1.4.1	03/25/2011	<ul style="list-style-type: none"> • Added explanation of Hermit-AT modes to Appendix A, Hermit-At Bootloader • Corrected part number errors in Section 9.1, “UART” • Added Section 6.7, “Restoring Bootloader Parameters to Factory State” • Made corrections to Section A.8.1, “setbootdevice Example” • Updated company address • Corrected various errors and inconsistencies • Updated Table 9.45, “Wakeup Basis Default Value Configuration” to include more detailed information • Added Section 9.20.1.1, “Power Supply to External Devices” • Updated Section 9.20.1.2, “Treatment of External Devices During Sleep” to include more detailed information • Updated Table 9.44, “Wakeup Basis Designation” for linux-2.6.26-at14
1.5.0	07/13/2011	<ul style="list-style-type: none"> • Added information on Armadillo-460 • Added Chapter 10, Armadillo-460 Expansion Bus
1.6.0	08/26/2011	<ul style="list-style-type: none"> • Added detailed descriptions about Table 3.3, “Default States of Armadillo-420 Basic Model Expansion Interfaces”, Table 3.5, “Default States of Armadillo-420 WLAN Model (AWL12 Compatible) Expansion Interfaces”, Table 3.9, “Default States of Armadillo-440 LCD Model Expansion Interfaces” and Table 3.10, “Default States of Armadillo-460 Basic Model Expansion Interfaces”. • Added CAN input as a wakeup basis in Table 9.43, “Sleep States” • Removed unsupported sampling frequencies from Section 9.8, “Audio” • Added recommended settings for serial communication software to Section 4.3, “Serial Console Software Configuration” • Corrected various errors
1.6.1	10/21/2011	<ul style="list-style-type: none"> • Updated web site name
1.7.0	12/21/2011	<ul style="list-style-type: none"> • Added descriptions of AWL13

Armadillo-400 Series Software Manual
Version 1.7.0
2012/02/29

Atmark Techno, Inc.

060-0035 AFT Bldg., N5E2, Chuo-ku, Sapporo TEL 011-207-6550 FAX 011-207-6570
