

Intel[®] EP80579 Software Drivers for Embedded Applications on Linux*

Getting Started Guide

November 2009

Order Number: 320151-007US



INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL® PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER, AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT. Intel products are not intended for use in medical, life saving, life sustaining, critical control or safety systems, or in nuclear facility applications.

Intel may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The information here is subject to change without notice. Do not finalize a design with this information.

The products described in this document may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

Copies of documents which have an order number and are referenced in this document, or other Intel literature, may be obtained by calling 1-800-548-4725, or by visiting Intel's Web Site.

Any software source code reprinted in this document is furnished under a software license and may only be used or copied in accordance with the terms of that license.

Intel processor numbers are not a measure of performance. Processor numbers differentiate features within each processor family, not across different processor families. See http://www.intel.com/products/processor_number for details.

BunnyPeople, Celeron, Celeron Inside, Centrino, Centrino logo, Core Inside, FlashFile, i960, InstantIP, Intel, Intel logo, Intel386, Intel486, Intel740, IntelDX2, IntelDX4, IntelSX2, Intel Core, Intel Inside, Intel Inside logo, Intel. Leap ahead., Intel. Leap ahead. logo, Intel NetBurst, Intel NetMerge, Intel NetStructure, Intel SingleDriver, Intel SpeedStep, Intel StrataFlash, Intel Viv, Intel vPro, Intel XScale, Itanium, Itanium Inside, MCS, MMX, Oplus, OverDrive, PDCharm, Pentium, Pentium Inside, skoool, Sound Mark, The Journey Inside, VTune, Xeon, and Xeon Inside are trademarks of Intel Corporation in the U.S. and other countries.

*Other names and brands may be claimed as the property of others.

Copyright © 2008-2009, Intel Corporation. All rights reserved.



Contents

1.0	Intro	oduction	6			
	1.1	.1 About this Manual				
	1.2	Additional Information on Software	6			
		1.2.1 Where to Find Current Software and Documentation	6			
		1.2.2 Product Documentation	6			
	1 3	Pelated Software and Documentation	/			
	1.3	Conventions	7			
	1.5	Software Overview	7			
		1.5.1 Features Implemented	7			
		1.5.2 List of Files in Release	7			
		1.5.3 Package Release Structure	7			
2.0	Conf	iguration Requirements	9			
	2.1	Development Board Configuration	9			
		2.1.1 Package Components	9			
		2.1.2 Development Kit Setup	9			
		2.1.3 Safety	. 10			
		2.1.4 Connecting the Serial ATA Hard Drive and Cable	. 10			
		2.1.5 Connecting the Keyboard and Mouse	. 10			
		2.1.6 Connecting the PCI Express Video Card	. 10			
		2.1.7 Connecting the Serial ATA DVD-ROM Drive (Optional)	. 11			
		2.1.8 Connecting the Power Cables	. 11			
	2.2	2.1.9 Powering up the System	. I I 15			
	2.2	2.2.1 Stand-alone Target System	15			
20	Such	m Requirements for Installing on OS on a Development Reard	14			
3.0	Syste	em Requirements for Installing an OS on a Development Board	. 16			
3.0	Syste 3.1	em Requirements for Installing an OS on a Development Board Acquiring CentOS 5.2 Linux	. 16 . 16 . 16			
3.0	Syste 3.1 3.2	Acquiring CentOS 5.2 Linux	. 16 . 16 . 16			
3.0	Syste 3.1 3.2 3.3	Acquiring CentOS 5.2 Linux Installing CentOS 5.2 Linux 3.2.1 Recommended Installation Customizations Unpacking the Software Package	. 16 . 16 . 16 . 17 . 18			
3.0	Syste 3.1 3.2 3.3 3.4	Acquiring CentOS 5.2 Linux Installing CentOS 5.2 Linux 3.2.1 Recommended Installation Customizations Unpacking the Software Package Updating PCI Device Names	. 16 . 16 . 16 . 17 . 18 . 18			
3.0	Syste 3.1 3.2 3.3 3.4 3.5	Acquiring CentOS 5.2 Linux Installing CentOS 5.2 Linux 3.2.1 Recommended Installation Customizations Unpacking the Software Package Updating PCI Device Names Build Environment Requirements	. 16 . 16 . 16 . 17 . 18 . 18 . 19			
3.0	Syste 3.1 3.2 3.3 3.4 3.5 Build	Acquiring CentOS 5.2 Linux Installing CentOS 5.2 Linux 3.2.1 Recommended Installation Customizations Unpacking the Software Package Updating PCI Device Names Build Environment Requirements	. 16 . 16 . 16 . 17 . 18 . 18 . 18 . 19 . 20			
3.0	Syste 3.1 3.2 3.3 3.4 3.5 Build 4.1	Acquiring CentOS 5.2 Linux Installing CentOS 5.2 Linux 3.2.1 Recommended Installation Customizations Unpacking the Software Package Updating PCI Device Names Build Environment Requirements Ling and Installing EP80579 Software on the Target Development Board Environment Setup.	. 16 . 16 . 17 . 18 . 18 . 18 . 19 . 20 . 20			
3.0	Syste 3.1 3.2 3.3 3.4 3.5 Build 4.1 4.2	Acquiring CentOS 5.2 Linux Installing CentOS 5.2 Linux 3.2.1 Recommended Installation Customizations Unpacking the Software Package Updating PCI Device Names Build Environment Requirements Iing and Installing EP80579 Software on the Target Development Board Environment Setup Build All Embedded Software Drivers	. 16 . 16 . 17 . 18 . 18 . 18 . 19 . 20 . 20 . 20			
3.0	Syste 3.1 3.2 3.3 3.4 3.5 Build 4.1 4.2 4.3	Acquiring CentOS 5.2 Linux Installing CentOS 5.2 Linux 3.2.1 Recommended Installation Customizations Unpacking the Software Package Updating PCI Device Names Build Environment Requirements Ing and Installing EP80579 Software on the Target Development Board Environment Setup Build All Embedded Software Drivers Uninstalling Embedded Software Drivers	. 16 . 16 . 16 . 17 . 18 . 18 . 18 . 19 . 20 . 20 . 20 . 21			
3.0 4.0 5.0	Syste 3.1 3.2 3.3 3.4 3.5 Build 4.1 4.2 4.3 Runt	Acquiring CentOS 5.2 Linux Installing CentOS 5.2 Linux Installing CentOS 5.2 Linux 3.2.1 Recommended Installation Customizations Unpacking the Software Package Updating PCI Device Names Build Environment Requirements Ing and Installing EP80579 Software on the Target Development Board Environment Setup Build All Embedded Software Drivers Uninstalling Embedded Software Drivers	. 16 . 16 . 17 . 18 . 18 . 19 . 20 . 20 . 20 . 21			
3.0 4.0 5.0	Syste 3.1 3.2 3.3 3.4 3.5 Build 4.1 4.2 4.3 Runt 5.1	Acquiring CentOS 5.2 Linux Installing CentOS 5.2 Linux 3.2.1 Recommended Installation Customizations Unpacking the Software Package Updating PCI Device Names Build Environment Requirements ling and Installing EP80579 Software on the Target Development Board Environment Setup Build All Embedded Software Drivers Uninstalling Embedded Software Drivers.	. 16 . 16 . 17 . 18 . 18 . 19 . 20 . 20 . 20 . 21 . 22			
3.0 4.0 5.0	Syste 3.1 3.2 3.3 3.4 3.5 Build 4.1 4.2 4.3 Runt 5.1	Acquiring CentOS 5.2 Linux Installing CentOS 5.2 Linux 3.2.1 Recommended Installation Customizations Unpacking the Software Package Updating PCI Device Names Build Environment Requirements Iing and Installing EP80579 Software on the Target Development Board Environment Setup Build All Embedded Software Drivers Uninstalling Embedded Software Drivers Uninstalling Embedded Software Drivers Software Drivers Ing Configuration Controller Area Network Driver. 5.1.1 Linux Compilation Instructions	. 16 . 16 . 17 . 18 . 17 . 18 . 19 . 20 . 20 . 20 . 21 . 22 . 22 . 22			
3.0 4.0 5.0	Syste 3.1 3.2 3.3 3.4 3.5 Build 4.1 4.2 4.3 Runt 5.1	Acquiring CentOS 5.2 Linux Installing CentOS 5.2 Linux 3.2.1 Recommended Installation Customizations Unpacking the Software Package Updating PCI Device Names Build Environment Requirements Iing and Installing EP80579 Software on the Target Development Board Environment Setup Build All Embedded Software Drivers Uninstalling Embedded Software Drivers Uninstalling Embedded Software Drivers S.1.1 Linux Compilation Instructions 5.1.2 Linux Module Load/Unload Instructions	. 16 . 16 . 17 . 18 . 18 . 19 . 20 . 20 . 20 . 20 . 21 . 22 . 22 . 22			
3.0 4.0 5.0	Syste 3.1 3.2 3.3 3.4 3.5 Build 4.1 4.2 4.3 Runt 5.1	em Requirements for Installing an OS on a Development Board. Acquiring CentOS 5.2 Linux Installing CentOS 5.2 Linux 3.2.1 Recommended Installation Customizations Unpacking the Software Package Updating PCI Device Names. Build Environment Requirements Iing and Installing EP80579 Software on the Target Development Board. Environment Setup. Build All Embedded Software Drivers Uninstalling Embedded Software Drivers. ime Configuration. Controller Area Network Driver. 5.1.1 Linux Compilation Instructions 5.1.2 Linux Module Load/Unload Instructions 5.1.3 Linux Runtime Configuration of CAN	. 16 . 16 . 16 . 17 . 18 . 18 . 19 . 20 . 20 . 20 . 20 . 22 . 22 . 22 . 22			
3.0 4.0 5.0	Syste 3.1 3.2 3.3 3.4 3.5 Build 4.1 4.2 4.3 Runt 5.1	em Requirements for Installing an OS on a Development Board. Acquiring CentOS 5.2 Linux Installing CentOS 5.2 Linux 3.2.1 Recommended Installation Customizations Unpacking the Software Package Updating PCI Device Names. Build Environment Requirements Iing and Installing EP80579 Software on the Target Development Board. Environment Setup. Build All Embedded Software Drivers Uninstalling Embedded Software Drivers. ime Configuration Controller Area Network Driver. 5.1.1 Linux Compilation Instructions 5.1.2 Linux Module Load/Unload Instructions 5.1.3 Linux Runtime Configuration of CAN 5.1.4 Linux Sample Codelet	. 16 . 16 . 16 . 17 . 18 . 18 . 19 . 20 . 20 . 20 . 20 . 22 . 22 . 22 . 22			
3.0 4.0 5.0	Syste 3.1 3.2 3.3 3.4 3.5 Build 4.1 4.2 4.3 Runt 5.1	em Requirements for Installing an OS on a Development Board. Acquiring CentOS 5.2 Linux Installing CentOS 5.2 Linux 3.2.1 Recommended Installation Customizations Unpacking the Software Package. Updating PCI Device Names. Build Environment Requirements. ling and Installing EP80579 Software on the Target Development Board. Environment Setup. Build All Embedded Software Drivers. Uninstalling Embedded Software Drivers. ime Configuration. Controller Area Network Driver. 5.1.1 Linux Compilation Instructions 5.1.2 Linux Module Load/Unload Instructions 5.1.3 Linux Runtime Configuration of CAN 5.1.4 Linux Sample Codelet Enhanced Direct Memory Access Driver	. 16 . 16 . 16 . 17 . 18 . 18 . 19 . 20 . 20 . 20 . 20 . 20 . 22 . 22 . 22			
3.0 4.0 5.0	Syste 3.1 3.2 3.3 3.4 3.5 Build 4.1 4.2 4.3 Runt 5.1	Acquiring CentOS 5.2 Linux Installing CentOS 5.2 Linux 3.2.1 Recommended Installation Customizations Unpacking the Software Package Updating PCI Device Names. Build Environment Requirements ling and Installing EP80579 Software on the Target Development Board Environment Setup. Build All Embedded Software Drivers Uninstalling Embedded Software Drivers. ime Configuration Controller Area Network Driver. 5.1.1 Linux Compilation Instructions 5.1.2 Linux Module Load/Unload Instructions 5.1.3 Linux Runtime Configuration of CAN 5.1.4 Linux Sample Codelet Enhanced Direct Memory Access Driver 5.2.1 Linux Compilation Instructions	. 16 . 16 . 16 . 17 . 18 . 18 . 19 . 20 . 20 . 20 . 20 . 20 . 20 . 22 . 22			
3.0 4.0 5.0	Syste 3.1 3.2 3.3 3.4 3.5 Build 4.1 4.2 4.3 Runt 5.1 5.2	Acquiring CentOS 5.2 Linux Installing CentOS 5.2 Linux 3.2.1 Recommended Installation Customizations Unpacking the Software Package Updating PCI Device Names. Build Environment Requirements ling and Installing EP80579 Software on the Target Development Board Environment Setup Build All Embedded Software Drivers Uninstalling Embedded Software Drivers Uninstalling Embedded Software Drivers S.1.1 Linux Compilation Instructions 5.1.2 Linux Module Load/Unload Instructions 5.1.4 Linux Sample Codelet Enhanced Direct Memory Access Driver 5.2.1 Linux Module Load/Unload Instructions 5.2.2 Linux Module Load/Unload Instructions 5.2.2 Linux Module Load/Unload Instructions 5.2.2 Linux Module Load/Unload Instructions	. 16 . 16 . 16 . 17 . 18 . 18 . 19 . 20 . 20 . 20 . 20 . 20 . 20 . 21 . 22 . 22 . 22 . 22 . 23 . 23 . 23 . 24			
3.0 4.0 5.0	Systa 3.1 3.2 3.3 3.4 3.5 Build 4.1 4.2 4.3 Runt 5.1	em Requirements for Installing an OS on a Development Board. Acquiring CentOS 5.2 Linux Installing CentOS 5.2 Linux 3.2.1 Recommended Installation Customizations Unpacking the Software Package Updating PCI Device Names. Build Environment Requirements Iing and Installing EP80579 Software on the Target Development Board. Environment Setup Build All Embedded Software Drivers Uninstalling Embedded Software Drivers. ime Configuration Controller Area Network Driver. 5.1.1 Linux Compilation Instructions 5.1.2 Linux Module Load/Unload Instructions 5.1.3 Linux Runtime Configuration of CAN 5.1.4 Linux Sample Codelet Enhanced Direct Memory Access Driver 5.2.1 Linux Runtime Configuration S 5.2.2 Linux Module Load/Unload Instructions 5.2.3 Linux Runtime Configuration of EDMA	. 16 . 16 . 17 . 18 . 19 . 20 . 20 . 20 . 20 . 20 . 20 . 20 . 20			
3.0 4.0 5.0	Syste 3.1 3.2 3.3 3.4 3.5 Build 4.1 4.2 4.3 Runt 5.1 5.2	em Requirements for Installing an OS on a Development Board. Acquiring CentOS 5.2 Linux Installing CentOS 5.2 Linux 3.2.1 Recommended Installation Customizations Unpacking the Software Package Updating PCI Device Names Build Environment Requirements Iing and Installing EP80579 Software on the Target Development Board Environment Setup Build All Embedded Software Drivers Uninstalling Embedded Software Drivers Uninstalling Embedded Software Drivers Softrant Controller Area Network Driver 5.1.1 Linux Compilation Instructions 5.1.2 Linux Module Load/Unload Instructions 5.1.3 Linux Runtime Configuration of CAN 5.1.4 Linux Sample Codelet Enhanced Direct Memory Access Driver 5.2.1 Linux Compilation Instructions 5.2.2 Linux Module Load/Unload Instructions 5.2.3 Linux Runtime Configuration of EDMA 5.2.4 Linux Sample Codelet	. 16 . 16 . 16 . 17 . 18 . 19 . 20 . 20 . 20 . 20 . 20 . 20 . 20 . 20			
3.0 4.0 5.0	Syste 3.1 3.2 3.3 3.4 3.5 Build 4.1 4.2 4.3 Runt 5.1 5.2	em Requirements for Installing an OS on a Development Board. Acquiring CentOS 5.2 Linux Installing CentOS 5.2 Linux 3.2.1 Recommended Installation Customizations Unpacking the Software Package Updating PCI Device Names. Build Environment Requirements ling and Installing EP80579 Software on the Target Development Board Environment Setup. Build All Embedded Software Drivers Uninstalling Embedded Software Drivers Uninstalling Embedded Software Drivers. ime Configuration. Controller Area Network Driver. 5.1.1 Linux Compilation Instructions 5.1.2 Linux Module Load/Unload Instructions 5.1.4 Linux Sample Codelet Enhanced Direct Memory Access Driver 5.2.1 Linux Runtime Configuration of EDMA 5.2.2 Linux Module Load/Unload Instructions 5.2.3 Linux Runtime Configuration of EDMA 5.2.4 Linux Sample Codelet CompactFlash Driver.	. 16 . 16 . 16 . 17 . 18 . 19 . 20 . 20 . 20 . 20 . 20 . 20 . 20 . 21 . 22 . 22 . 22 . 22 . 23 . 23 . 23 . 23			



	5.4	5.3.2 Linux Module Load/Unload Instructions 25 5.3.3 Linux Runtime Configuration of CompactFlash Driver 25 VDT - Watchdog Timer 26 5.4.1 Linux Compilation Instructions 26 5.4.2 Linux Module Load/Unload Instructions 26 5.4.3 Linux Runtime Configuration of WDT 26 5.4.4 Linux Runtime Configuration of WDT 26	
	5.5	5.4.4 Linux Sample Codelet 26 GPIO 27 5.5.1 Linux Compilation Instructions 27 5.5.2 Linux Module Load/Unload Instructions 27 5.5.3 Linux Runtime Configuration of GPIO 27) , , ,
	5.6	EEE 1588 Hardware Assist 28 5.6.1 Linux Compilation Instructions 28 5.6.2 Linux Module Load/Unload Instructions 28 5.6.3 Linux Runtime Configuration of IEEE 1588 28 5.6.4 Linux Sample Codelet 28	
	5.7	Global Configuration Unit and Gigabit Ethernet Driver 29 5.7.1 Linux Compilation Instructions 29 5.7.2 Linux Module Load/Unload Instructions 29 5.7.3 Linux Runtime Configuration of GCU and Gigabit Ethernet 30	, , ,)
	5.8	SMBus 30 5.8.1 Linux Compilation Instructions 30 5.8.2 Linux Module Load/Unload Instructions 30 5.8.3 Linux Runtime Configuration of SMBus 31))
6.0	Booti	g from CompactFlash* 32	,
	6.1 6.2	System Requirements))
7.0	Pre-b	ot (BIOS) Firmware	,
	7.1	Pre-boot Firmware Setup Menu) , , , , , , , , , , , , ,
	7.2	7.1.5 Legacy and AHCI SATA Mode	;))
8.0	Unins	alling the Software41	I
	8.1	inux Module/Driver Dependencies	
9.0	Trout	eshooting)
10.0	Glossary		

Figures

<u> </u>		
1	Software Package Release Structure - Linux	. 8
2	Development Board - Component and Connector Locations	12
3	Side View of the Board Connectors	13
4	Development Board System Setup	15

Tables

1	Key Development Board Components and Connectors Legend	13
2	Pre-boot Firmware Setup Main Menu	36



3	Pre-boot Firmware Setup Program Function Keys	. 36
4	Serial Console Redirection Default Settings	. 37
5	Memory Allocation Settings	. 38
6	Linux Module/Driver Dependencies	. 41

Revision History

Date	Revision	Description
November 2009	007	Minor modifications to correct URLs. Change bars were not updated; change bars show edits from previous doc version (below).
September 2009	006	Added Note in Section 5.1.4, "Linux Sample Codelet" on page 23. Added Section 6.0, "Booting from CompactFlash*" on page 32. Other updates noted with changebars.
May 2009	005	Minor modifications to correct URLs and other text corrections. Change bars were not updated; change bars show edits from previous doc version (below).
March 2009	004	Updated directory structure in Figure 1. Added Section 5.1.4, "Linux Sample Codelet" on page 23 for CAN. Added Section 5.4.4, "Linux Sample Codelet" on page 26 for WDT. Added Section 5.6.4, "Linux Sample Codelet" on page 28 for IEEE 1588.
November 2008	003	General updates to reflect a change of OS from Red Hat* to CentOS. Updated directory structure in Figure 1. Updated In command in Section 4.1, "Environment Setup" on page 20. Added Section 5.2.4, "Linux Sample Codelet" on page 24.
September 2008	002	Added information for alternate PHY support (see Section 9.6.4, "Alternative PHY Installation/ Compilation" on page 59).
July 2008	001	Initial release of this document.

§§



1.0 Introduction

1.1 About this Manual

This Getting Started Guide documents the instructions to obtain, build (if necessary), install, and execute the software release package for the Intel[®] EP80579 Integrated Processor product line. Additionally, this document describes brief installation instructions for the Intel[®] EP80579 Integrated Processor with Intel[®] QuickAssist Technology Development Board.

Note: The "Intel[®] EP80579 Integrated Processor with Intel[®] QuickAssist Technology Development Board" will be referred to as the "development board" in this document.

1.2 Additional Information on Software

The Intel[®] EP80579 Software Drivers for Embedded Applications on Linux* package has been validated with CentOS 5.2 Linux.

1.2.1 Where to Find Current Software and Documentation

The software release and associated collateral can be found on the Hardware Design resource center.

- 1. In a web browser, go to http://www.intel.com/go/soc.
- 2. For Software and Pre-boot Firmware: Click on "Tools & Software" tab.
- 3. For Documentation: Click on "Technical Documents" tab.

1.2.2 Product Documentation

The following documentation is provided to support this software release:

- This Getting Started Guide
- Intel[®] EP80579 Software Drivers for Embedded Applications Release Notes
- Intel[®] EP80579 Software Drivers for Embedded Applications Programmer's Guide and API Reference Manual
- Software for Intel[®] EP80579 Integrated Processor Product Line PHY Porting Guide
- *Note:* The Release Notes contains important information about each software release, such as the appropriate firmware version. Please review the Release Notes before proceeding with this document.

Please follow the directions in Section 1.2.1 to locate this collateral.



1.2.3 Pre-boot Firmware

The latest release of the development board pre-boot firmware (BIOS) is also located on Hardware Design resource center. Refer to the Intel[®] EP80579 Software Drivers for Embedded Applications Release Notes for latest version.

Please follow the directions in Section 1.2.1 to locate this firmware.

1.3 Related Software and Documentation

Refer to the Intel[®] EP80579 Integrated Processor with Intel[®] QuickAssist Technology Development Kit User's Guide for information on the development board including board layout, components, connectors, jumpers, headers, power and environmental requirements, and pre-boot firmware.

Please follow the directions in Section 1.2.1 to locate this collateral.

1.4 Conventions

The following conventions are used in this manual:

- Courier font commands and code examples
- · Italics directory names

1.5 Software Overview

1.5.1 Features Implemented

The software provides the following features:

- IEEE 1588 Hardware Assist Driver
- Controller Area Network (CAN) Hardware Access Driver
- Gigabit Ethernet (GbE) Controller Driver for Network Connectivity
- Advanced Host Controller Interface Software Support for SATA for Native Command Queuing and Hot Plug Capability
- CompactFlash Driver
- SMBus Driver
- General Purpose I/O (GPIO) Hardware Access Driver
- Enhanced Direct Memory Access (EDMA) Hardware Assist Driver
- Watchdog Timer Hardware (WDT) Access Driver

1.5.2 List of Files in Release

The Bill of Materials, sometimes referred to as the BOM, is included as a text file in the released software package. This text file is labeled "filelist" and is located at the top directory level for each release.

1.5.3 Package Release Structure

The package release structure is shown in Figure 1.



Figure 1. Software Package Release Structure - Linux





2.0 Configuration Requirements

2.1 Development Board Configuration

Complete details about the development board can be found in the "Intel[®] EP80579 Integrated Processor with Intel[®] QuickAssist Technology Development Kit User's Guide". This document contains details on the design, structure, and function of all development board features.

To facilitate quick start of the EP80579 software drivers for embedded applications package, relevant sections from the Development Kit User's Guide have been included in this chapter. Please follow the directions in Section 1.2.1 for information on accessing the full User's Guide.

2.1.1 Package Components

The Intel[®] EP80579 Integrated Processor with Intel[®] QuickAssist Technology Development Kit includes the following:

- Development board containing the Intel[®] EP80579 Integrated Processor with Intel[®] QuickAssist Technology
- ATX12V power supply
- One DDR2-800 DIMM
- PCIe* graphics card
- SATA hard drive with cable
- SATA DVD-ROM with cable
- Two Controller Area Network cable connectors
- Power Cord (USA power cord supplied)

These items are not supplied from Intel. Please note this is not an exhaustive list of items not supplied.

- Mouse
- Keyboard
- Monitor
- Power supply cord (if country or region-specific power cord is required)

2.1.2 Development Kit Setup

Ensure that all components listed in Section 2.1.1 arrive together. Once all components have been identified and located, installation and setup can begin. This section describes how to set up the development board for operation.

Note: This document assumes that the user is familiar with the basic concepts required to install and configure hardware for a PC system.



2.1.3 Safety

The development board is shipped as an open system allowing for maximum flexibility in changing hardware configurations and peripherals in a lab environment. Since the board is not in a protective chassis, the user is required to take safety precautions in handling and operating the board. Some assembly is required before use.

Ensure a safe and static-free work environment before removing any components from their anti-static packaging. The development board is susceptible to electrostatic discharge that may cause failure or unpredictable operation. The development board must be operated on a flame-retardant surface because a chassis is not included with the board.

- *Caution:* Connecting the wrong cable or reversing a cable may damage the board and may damage the device being connected. Since the board is not in a protective chassis, use caution when connecting cables to the board.
- *Caution:* The power supply cord is the main disconnect device to main power (AC power). The socket outlet should be installed near the equipment and should be readily accessible. To avoid shock, ensure that the power cord is connected to a properly wired and grounded receptacle. Do not connect/disconnect any cables or perform installation/ maintenance of the boards in this product during an electrical storm. Ensure that any equipment to which this product will be attached is also connected to properly wired and grounded receptacles.
- *Note:* Ensure that setting up the ATX power supply is the final step performed in the process of assembly.

2.1.4 Connecting the Serial ATA Hard Drive and Cable

The development board provides two Serial ATA (SATA) connectors. Connect cables to the appropriate drive sequentially, starting from Port 0 to Port 1. See Figure 2 and Table 1 for the location and identification of the SATA connectors.

Note: Intel recommends connecting the boot drive to SATA port 0.

2.1.5 Connecting the Keyboard and Mouse

Connect a PS/2 mouse and keyboard to the stacked PS/2 connector on the rear panel of the board. The bottom connector is the keyboard connector and the top connector is the mouse connector. Alternatively, a USB keyboard and a USB mouse can be connected to the development board's USB connectors.

- *Note:* Mouse and keyboard are not supplied by Intel.
- *Note:* The serial redirection feature can be enabled to remotely access the board through a serial cable without attaching a keyboard or mouse to the development board. Refer to the "Connecting the Serial Cable for Console Redirection" section of the Intel[®] EP80579 Integrated Processor with Intel[®] QuickAssist Technology Development Kit User's Guide for more information.

2.1.6 Connecting the PCI Express Video Card

Populate the PCIe graphics card in any one of the PCIe slots.



2.1.7 Connecting the Serial ATA DVD-ROM Drive (Optional)

Connect the Serial ATA DVD-ROM drive to SATA Port 1 utilizing the cable that comes with the DVD-ROM drive. See Figure 2 and Table 1 for the location and identification of the SATA connectors.

2.1.8 Connecting the Power Cables

Use the following procedure to connect the power cables:

- 1. The board supports the use of ATX12V power supplies with either 2 x 10 or 2 x 12 main power cables.
- 2. Plug the main connector into the board. Ensure that the plug clip lines up with the clip lock and the connector pins easily fit into their appropriate slots. When using a power supply with a 2 x 10 main power cable, attach that cable to the right-most part of the main power connector, leaving pins 11, 12, 23 and 24 unconnected.
- 3. Plug in the power connectors from each of the SATA drives and disk drives.

2.1.9 Powering Up the System

Warning: Ensure the steps in the previous sections were strictly followed before powering up the system.

Use the following procedures to power up the development board:

- Ensure that the processor heat sink and the fan are installed according to the procedure in the "Connecting the Processor Heatsink and Fan" section of the Intel[®] EP80579 Integrated Processor with Intel[®] QuickAssist Technology Development Board User's Guide.
- 2. Leaving the On/Off switch in the OFF position, plug the power cable into the back of the power supply.
- 3. Once the board is set up, plug the cord into the power source.
- 4. Switch on the power supply.
- 5. Press the power-on button to start the system. Refer to Figure 2 for the location power-on button (item I, lower-left). Table 1 is a legend for key items labeled in Figure 2.



DD CC BB AA	Y X	C G F
	n Min	
s 🛶 🖬 👘 👘 👘		
7		
R		
s		
	01.001.008 23 14 0 <u>8 9 100</u> 1	
8 8 4 6	KC ADDRE & LABEL 2	
**************************************	AC ADDATS LABEL 3	
5 Wi 5 V - K		- III IIII

Figure 2. Development Board - Component and Connector Locations

B6607-01



Figure 3. Side View of the Board Connectors



B6605-01

Callout	Component/Connector
A	Intel [®] EP80579 Integrated Processor
В	PEX PCIe Switch Chip
С	Marvell 8811E1 Quad PHY
D	Super IO Controller
E	FPGA
F	Flash memory 0
G	Flash memory 1
Н	FWH
I	Power button
J	Reset button
К	Sleep button
L	PCIe Wake button
М	Port 80 IC
N	CMOS battery
0	On-board speaker
Р	CPU FAN connector
Q	AUX FAN connector
R	AUX0 FAN connector
S	AUX1 FAN connector
Т	ATX power connector
U	Two 7-segment display (Port 80)
V	SATA port 0
W	SATA port 1

Table 1. Key Development Board Components and Connectors Legend

Callout	Component/Connector
Х	DDR2 DIMM0
Y	DDR2 DIMM1
Z	Slot 0 x8 connector 4 lanes PCI Express
AA	Slot 1 x4 connector 1 lane PCI Express
BB	Slot 2 x4 connector 1 lane PCI Express
CC	Slot 3 x4 connector 1 lane PCI Express
DD	Slot 4 x4 connector 1 lane PCI Express
EE	Mezzanine connector 0
FF	Mezzanine connector 1
GG	Floppy Connector
HH	CF connector
П	ITP-XDP connector
LL	Mezzanine connector 3
MM	Parallel port
NN	COM1
00	COM2
PP	PS/2 mouse (top)/keyboard (bottom)
QQ	USB port 0
RR	USB port 1
SS	RJ-45 Ethernet port 0
TT	RJ-45 Ethernet port 1
UU	RJ-45 Ethernet port 2
Note: Items JJ intentionally or	and KK are not shown in Figure 2 or Figure 3 and are nitted.

Table 1. Key Development Board Components and Connectors Legend



2.2 Development Board Setup Requirements

2.2.1 Stand-alone Target System

Figure 4 shows the system setup when the target development board is also used for build and install.

Figure 4. Development Board System Setup





3.0 System Requirements for Installing an OS on a Development Board

Note: Please consult the CentOS Linux* minimum hardware requirements.

3.1 Acquiring CentOS 5.2 Linux

The software package has been validated with CentOS 5.2, and validated with GCC 4.1 and the Glibc 2.5 tool chain. The software package does not include a distribution of CentOS 5.2 or any Linux distribution. The package includes Linux device driver source developed by Intel for EP80579 integrated processor features. The device driver source may be installed with a CentOS 5.2 distribution. Please acquire a source distribution of CentOS 5.2 Linux from http://vault.centos.org/5.2.

3.2 Installing CentOS 5.2 Linux

Note: User will be presented an option to install Linux in either text or graphical mode. The instructions in this guide are based with graphical mode selected. Installation instructions may differ if text based installation is selected.

- 1. For complete CentOS installation instructions, please refer to the online Installation Guide at http://www.centos.org/docs/5/html/5.2/Installation_Guide.The following are some basic instructions. For the purposes of this Getting Started Guide, it is assumed that the installation is from CD images. Prepare the system to boot from CDROM by using the boot selections in the BIOS Setup Menu. Details of the BIOS setup options are available in Chapter 7.0, "Pre-boot (BIOS) Firmware".
- 2. Ensure disc #1 is inserted into the CDROM prior to powering on the system.
- 3. Power on the system with disc #1 in the CDROM. The system should begin to boot from the CentOS installation disc #1 in the CDROM.
- *Note:* It is recommended that all CentOS installation discs be verified at least once prior to an installation of the OS.
 - 4. Upon booting from disc #1, the installation process prompts if the CD media should be tested.
 - 5. Review the CentOS Release Notes during the installation if desired. Continue with the installation process whenever prompted with the "Next" button.
 - 6. Select the language to use during the CentOS installation process.
 - 7. Select the appropriate keyboard for the system.
 - 8. Select option to perform fresh install of CentOS on the system.
 - 9. The hard drive that comes with the development board is initially blank. Partition this hard drive.
 - 10. Select the region this system is located in.
 - 11. Select the nearest city this system is located to.



- 12. Enter the desired root password for the system in the box labeled "Root Password:" and reaffirm the root password by entering the password in the box labelled "Confirm:".
- 13. The user is presented with a number of default installation application sets including Desktop Gnome, Desktop KDE, Server, and so on. Desktop Gnome is selected by default. Add other configurations as needed.

IMPORTANT: You must select the "Customize Now" button in step 14. to get access to the "Software Development" options required to complete the installation successfully.

14. Select "Customize Now" and see Section 3.2.1, which provides "Development" options that enable GNU Compiler Collection (Gcc) and GlibC Tool Set installation. Select the desired sets of software to include. Selecting the "Software Development" software set ensures that Gcc and GLibC Tool Set are installed. These are necessary to allow the compilation of the EP80579 embedded software drivers and the rebuilding of the Linux kernel to support the EP80579 integrated processor.

3.2.1 Recommended Installation Customizations

- 1. Package customization is available through these selections:
- Desktop Environments
- Applications
- Development
- Servers
- Base System
- Language

The recommended minimum package installation selections are given in the tips below. Additional detailed customization of each package can be accomplished by selecting a package (example: Editor), and clicking the "Optional package" button to display the optional and default packages within the main package.

- *Tip:* **Desktop Environments:** Either GNOME or the KDE Desktop Environment is suitable.
- *Tip:* **Applications:** Editor, Graphical Internet, and Text-based Internet are recommended to be installed.
- *Tip:* **Development:** Development Libraries and Development Tools are recommended to be installed.
- *Tip:* **Servers:** Depending on the end use connectivity, none to all server types may apply. Some common selections are DNS Name Server, FTP Server, Network Servers or Web Servers.
- *Tip:* **Base System:** Administration Tools, Base, System Tools, and X Windows System are recommended to be installed.
- *Tip:* Languages: Additional language selections vary depending on user location.
 - 2. Following package selections, click the "Next" button when prompted. After a dependency check, the installation process is ready to begin and prompts the user to have the required CentOS CDs available. To begin the installation, click the "Continue" button, click "Back" to change a package selection, or click "Reboot" to abort this installation.



- 3. If continuing with this installation, when prompted, insert the requested installation CD and click "Next". At installation complete, when prompted to reboot the system, remove the final CD from the CDROM and choose to reboot the system.
- 4. When rebooting, some normal system setup is required, such as setting the time and date. When asked, follow the prompt by clicking the "Forward" button.
- 5. Select "Trusted services" under Firewall settings. The recommended minimums are FTP, SSH, and Telnet. Click "Yes" to confirm security level settings overriding existing firewall configuration.
- 6. SELinux may be either enforced or disabled.
- 7. Set the time and date.
- 8. Finally, when prompted, press the "Finish" button.

3.3 Unpacking the Software Package

The EP80579 embedded software drivers package comes in the form of a tarball. See Section 1.2.1, "Where to Find Current Software and Documentation" on page 6 for the software location. The package can be unpacked at any location on the system, but for the purposes of this Getting Started Guide, a recommendation is provided.

Create a directory in the root directory called "EP805XX_release" with the following commands:

```
cd /
mkdir EP805XX_release
cd EP805XX release
```

Transfer the tarball to the development board using any preferred method, for example, USB memory stick, CDROM or network transfer. Place and unpack the tarball in the */EP805XX_release* directory using the following command:

```
tar -xvzf <tarball name>
```

A new directory structure is created under the *\$ICP_ROOT* directory that contains all EP80579 embedded software drivers for Linux. See Figure 1, "Software Package Release Structure - Linux" on page 8 for a detailed view of the directory structure of the EP80579 software drivers for embedded applications release.

3.4 Updating PCI Device Names

PCI device names must be updated prior to driver installation. This can be done by performing the following steps:

- Download the most recent PCI ID list from: http://pciids.sourceforge.net/pci.ids.bz2 mv /usr/share/hwdata/pci.ids /usr/share/hwdata/pci.ids.old
- 2. Extract the contents of the archive bunzip2 pci.ids.bz2
- Copy the updated file into the /usr/share/hwdata directory mv pci.ids /usr/share/hwdata

The utility "Ispci" can be used to verify the changes. The strings returned by the command will contain "Intel Corporation EP80579".

System Requirements for Installing an OS on a Development Board—EP80579 Integrated Processor



3.5 Build Environment Requirements

CentOS 5.2 includes GCC 4.1.1 and the Glibc 2.5 tool chain.

The CentOS 5.2 "Software Development" package should be selected at install time to install GCC 4.1 and the Glibc 2.5 tool chain.



4.0 Building and Installing EP80579 Software on the Target Development Board

This chapter provides instructions for setting up the environment and building/ compilation instructions for the Intel[®] EP80579 Software Drivers for Embedded Applications package.

4.1 Environment Setup

After unpacking the EP80579 embedded software drivers package, two environment variables must be set and one symbolic link must be created using the following commands:

export ICP_ENV_DIR=/EP805XX_release/Embedded
export ICP_ROOT=/EP805XX_release
mkdir -p /usr/src/kernels

ln -s /usr/src/kernels/2.6.18-92.el5-i686 /usr/src/kernels/linux

Note: The first two export commands need to be performed in each new shell. The final command to create a symbolic linked directory only needs to be performed once.

Note: If new kernel sources are to be used, the symbolic link must be updated to point to the new kernel source directory.

4.2 Build All Embedded Software Drivers

Change directory to the *\$ICP_ROOT/Embedded* directory using the following command:

cd \$ICP ROOT/Embedded

Build all software drivers within the package using the following command:

make clean make

When executing the make command from the *\$ICP_ROOT/Embedded* directory, a *\$ICP_ROOT/Embedded/build* directory is created and copies of all component compiled binaries are placed in this directory.

To install the drivers for persistence on future boot, execute the following command from the *\$ICP_ROOT/Embedded* directory:

make install

Note: The GPIO driver is not persistent after the make install command is issued. This driver is included to illustrate the usage of GPIO.



Under the *\$ICP_ROOT/Embedded/src* directory are directories for each feature component. Within each feature component directory, a *build/linux_2.6/kernel_space* directory is created containing the compiled executable for that feature. For example, the compiled executable for the EDMA driver resides in the *\$ICP_ROOT/Embedded/src/EDMA/build/linux_2.6/kernel_space* directory.

At this point installation of the drivers is complete.

Note: Instructions for building and installing drivers individually are also provided in this guide. See Chapter 5.0, "Runtime Configuration".

See the Software for Intel[®] EP80579 Integrated Processor Product Line PHY Porting Guide for information on porting the driver to other PHYs.

4.3 Uninstalling Embedded Software Drivers

The make uninstall command is available to uninstall all drivers from the top level *\$ICP_ROOT/Embedded* directory. A reboot is required to complete the uninstall process. To uninstall all the EP80579 embedded software drivers, execute the following command from the *\$ICP_ROOT/Embedded* directory:

make uninstall shutdown -r now

§§



5.0 Runtime Configuration

This chapter describes how to compile and install EP80579 drivers individually.

5.1 Controller Area Network Driver

5.1.1 Linux Compilation Instructions

All source files for the Linux* release of the Controller Area Network (CAN) driver are located in the following directory within the Linux compatible EP80579 embedded software drivers release:

\$ICP ROOT/Embedded/src/CAN

Compilation of the Linux CAN driver separately from the rest of the software package is possible. Change to the *\$ICP_ROOT/Embedded/src/CAN* directory and execute the following commands:

make clean make

The CAN driver compiles and the resulting can.ko file is placed in the *\$ICP_ROOT/Embedded/src/CAN/build/linux_2.6/kernel_space* directory.

5.1.2 Linux Module Load/Unload Instructions

To load the Linux CAN driver, execute the following command from the directory where the compiled executable resides:

insmod can.ko

The driver can also be installed for persistency using the "make install" command from the *\$ICP_ROOT/Embedded/src/CAN* directory.

To unload the Linux CAN driver execute the following command:

rmmod can.ko

The Ismod command may be used to confirm if a module has been loaded or unloaded:

lsmod | grep can

The output of this Ismod command lists modules loaded in the system containing "can" as part of their name.



5.1.3 Linux Runtime Configuration of CAN

Runtime configuration of the Controller Area Network driver is performed from a client driver communicating through the published API set described in the Intel[®] EP80579 Software Drivers for Embedded Applications Programmer's Guide and API Reference Manual. Intel leaves the design and development of a client driver to the customer since it is implementation specific.

5.1.4 Linux Sample Codelet

A codelet is provided to demonstrate how a sample application interfaces with the Controller Area Network driver. This codelet is intended to run on an Intel[®] EP80579 Integrated Processor with Intel[®] QuickAssist Technology Development Board with CentOS installed. The EP80579 Embedded Software kernel should also be installed, but the codelet works even if it is not. Run the codelet as follows:

Note:

In order to run this application, make sure a loopback CAN cable is connected to CAN 0 and CAN 1 port through CAN header. For additional details, refer to the Readme document location in */EP805XX_release/Embedded/codelet/CAN*.

- 1. Login as root. Root permissions are required for all operations.
- 2. Open a terminal window, change directory to *\$ICP_ROOT/Embedded/codelet/CAN* and execute the bash script. Do not move the script to another location.

[root@localhost ~]# ./install.bash

The script checks for the Controller Area Network driver and client drivers, runs all of the makefiles in the correct order, loads the client driver, and runs the application.

The application output can be viewed in the terminal window.

3. Client driver output can be viewed in a second terminal window by printing /var/log/messages as follows:

[root@localhost ~]# tail -f /var/log/messages

4. For information about the CAN codelet, refer to the Readme document located at *\$ICP_ROOT/Embedded/codelet/CAN*.

5.2 Enhanced Direct Memory Access Driver

5.2.1 Linux Compilation Instructions

All source files for the Linux release of the Enhanced Direct Memory Access (EDMA) driver are located in the following directory within the Linux compatible EP80579 embedded software drivers release:

\$ICP_ROOT/Embedded/src/EDMA

Compilation of the Linux EDMA driver separately from the rest of the software package is possible. Change to the *\$ICP_ROOT/Embedded/src/EDMA* directory and execute the following commands:

make clean make

The EDMA driver compiles and the resulting dma.ko file is placed in the *\$ICP_ROOT/Embedded/src/EDMA/build/linux_2.6/kernel_space* directory.



5.2.2 Linux Module Load/Unload Instructions

To load the Linux EDMA driver, execute the following command from the directory where the compiled executable resides:

insmod dma.ko

Note: The driver can also be installed for persistency using the "make install" command from the *\$ICP_ROOT/Embedded/src/EDMA* directory.

To unload the Linux EDMA driver, execute the following command:

rmmod dma.ko

The Ismod command may be used to confirm if a module has been loaded or unloaded:

lsmod | grep dma

The output of this Ismod command lists modules loaded in the system containing "dma" as part of their name.

5.2.3 Linux Runtime Configuration of EDMA

Runtime configuration of the Enhanced Direct Memory Access driver is performed from a client driver communicating through the published API set described in the Intel[®] EP80579 Software Drivers for Embedded Applications Programmer's Guide and API Reference Manual. Intel leaves the design and development of a client driver to the customer since it is implementation specific.

5.2.4 Linux Sample Codelet

A codelet is provided to demonstrate how a sample application interfaces with the EDMA driver. This codelet is intended to run on an Intel[®] EP80579 Integrated Processor with Intel[®] QuickAssist Technology Development Board with CentOS installed. The EP80579 Embedded Software kernel should also be installed, but the codelet works even if it is not. Run the codelet as follows:

- 1. Login as root. Root permissions are required for all operations.
- 2. Open a terminal window, change directory to *\$ICP_ROOT/Embedded/codelet/EDMA* and execute the bash script. Do not move the script to another location.

[root@localhost ~]# ./install.bash

The script checks for the EDMA and client drivers, runs all of the makefiles in the correct order, loads the client driver, and runs the application. The application output can be viewed in the terminal window.

3. Client driver output can be viewed in a second terminal window by printing /var/log/messages as follows:

[root@localhost ~]# tail -f /var/log/messages

4. For information about the EDMA codelet, refer to the Readme document located at \$ICP_ROOT/Embedded/codelet/EDMA.



5.3 CompactFlash Driver

5.3.1 Linux Compilation Instructions

All source files for the Linux release of the CompactFlash (CF) driver are located in the following directory within the Linux compatible EP80579 embedded software drivers release:

\$ICP ROOT/Embedded/src/CF

Compilation of the Linux CFdriver separately from the rest of the software package is possible. Change to the *\$ICP_ROOT/Embedded/src/CF* directory and execute the following commands:

make clean make

The CF driver compiles and the resulting leb_cf.ko file is placed in the *\$ICP_ROOT/Embedded/src/CF/build/linux_2.6/kernel_space* directory.

5.3.2 Linux Module Load/Unload Instructions

To load the Linux CF driver, execute the following command from the directory where the compiled executable resides:

insmod leb cf.ko

Note: The driver can also be installed for persistency using the "make install" command from the *\$ICP_ROOT/Embedded/src/CF* directory.

To unload the Linux CFdriver, execute the following command:

rmmod leb_cf.ko

The Ismod command may be used to confirm if a module has been loaded or unloaded:

lsmod | grep leb cf

The output of this Ismod command lists modules loaded in the system containing "leb_cf" as part of their name.

5.3.3 Linux Runtime Configuration of CompactFlash Driver

No runtime configuration is necessary for the CompactFlash driver. Once the component is built and installed, the driver will be running. Note that Compact Flash card must be mounted in order to use as storage medium, using the following commands:

mkdir /mnt/cf
mount /dev/sda1 /mnt/cf

The development platform can be configured to boot from Compact Flash. For additional information refer to Section 6.0, "Booting from CompactFlash*" on page 32.



5.4 WDT - Watchdog Timer

5.4.1 Linux Compilation Instructions

All source files for the Linux release of the Watchdog Timer (WDT) driver are located in the following directory within the Linux compatible EP80579 embedded software drivers release:

\$ICP ROOT/Embedded/src/WDT

Compilation of the Linux WDT driver separately from the rest of the software package is possible. Change to the *\$ICP_ROOT/Embedded/src/WDT* directory and execute the following commands:

make clean make

The Watchdog Timer driver compiles and the resulting wdt.ko file is placed in the *\$ICP_ROOT/Embedded/src/WDT/build/linux_2.6/kernel_space* directory.

5.4.2 Linux Module Load/Unload Instructions

To load the Linux Watchdog Timer driver, execute the following command from the directory where the compiled executable resides (*\$ICP_ROOT/Embedded/src/WDT/build/linux_2.6/kernel_space*):

insmod wdt.ko

Note: The driver can also be installed for persistency using the "make install" command from the *\$ICP_ROOT/Embedded/src/WDT* directory.

To unload the Linux Watchdog Timer driver, execute the following command:

rmmod wdt.ko

The Ismod command may be used to confirm if a module has been loaded or unloaded:

lsmod | grep wdt

The output of this Ismod command lists modules loaded in the system containing "wdt" as part of their name.

5.4.3 Linux Runtime Configuration of WDT

Runtime configuration of the Watchdog Timer driver is performed from a client driver communicating through the published API set described in the Intel[®] EP80579 Software Drivers for Embedded Applications Programmer's Guide and API Reference Manual. Intel leaves the design and development of a client driver to the customer since it is implementation specific.

5.4.4 Linux Sample Codelet

A codelet is provided to demonstrate how a sample application interfaces with the Watchdog Timer driver. This codelet is intended to run on an Intel[®] EP80579 Integrated Processor with Intel[®] QuickAssist Technology Development Board with CentOS installed. The EP80579 Embedded Software kernel should also be installed, but the codelet works even if it is not. Run the codelet as follows:

1. Login as root. Root permissions are required for all operations.



2. Open a terminal window, change directory to *\$ICP_ROOT/Embedded/codelet/WDT* and execute the bash script. Do not move the script to another location.

[root@localhost ~]# ./install.bash

The script checks that the Watchdog Timer driver is loaded, runs all of the makefiles in the correct order, and runs the application. The application output can be viewed in the terminal window.

3. For information about the WDT codelet, refer to the Readme document located at \$ICP_ROOT/Embedded/codelet/WDT.

5.5 **GPIO**

5.5.1 Linux Compilation Instructions

All source files for the Linux release of General Purpose I/O (GPIO) driver are located in the following directory within the Linux compatible EP80579 embedded software drivers release:

\$ICP ROOT/Embedded/src/GPIO

Compilation of the Linux GPIO driver separately from the rest of the software package is possible. Change to the *\$ICP_ROOT/Embedded/src/GPIO* directory and execute the following commands:

make clean make

The GPIO driver compiles and the resulting gpio.ko file is placed in the *\$ICP_ROOT/Embedded/src/GPIO/build/linux_2.6/kernel_space* directory.

5.5.2 Linux Module Load/Unload Instructions

To load the Linux General Purpose I/O driver, execute the following command from the directory where the compiled executable resides:

insmod gpio.ko

To unload the Linux General Purpose I/O driver, execute the following command:

rmmod gpio.ko

The Ismod command may be used to confirm if a module has been loaded or unloaded:

lsmod | grep gpio

The output of this Ismod command lists modules loaded in the system containing "gpio" as part of their name.

5.5.3 Linux Runtime Configuration of GPIO

Runtime configuration of the General Purpose I/O driver is performed from a client driver communicating through the published API set described in the Intel[®] EP80579 Software Drivers for Embedded Applications Programmer's Guide and API Reference Manual. Intel leaves the design and development of a client driver to the customer since it is implementation specific.



5.6 IEEE 1588 Hardware Assist

5.6.1 Linux Compilation Instructions

All source files for the Linux release of the IEEE 1588 Hardware Assist (1588) driver are located in the following directory within the Linux compatible EP80579 embedded software drivers release:

\$ICP ROOT/Embedded/src/1588

Compilation of the Linux IEEE 1588 driver separately from the rest of the software package is possible. Change to the *\$ICP_ROOT/Embedded/src/1588* directory and execute the following commands:

make clean make

The IEEE 1588 Hardware Assist driver compiles and the resulting timesync.ko file is placed in the *\$ICP_ROOT/Embedded/src/1588/build/linux_2.6/kernel_space* directory.

5.6.2 Linux Module Load/Unload Instructions

To load the Linux IEEE 1588 Hardware Assist driver, execute the following command from the directory where the compiled executable resides:

insmod timesync.ko

Note: The driver can also be installed for persistency using the "make install" command from the *\$ICP_ROOT/Embedded/src/1588* directory.

To unload the Linux IEEE 1588 Hardware Assist driver, execute the following command:

rmmod timesync.ko

The Ismod command may be used to confirm if a module has been loaded or unloaded:

lsmod | grep timesync

The output of this Ismod command lists modules loaded in the system containing "timesync" as part of their name.

5.6.3 Linux Runtime Configuration of IEEE 1588

Runtime configuration of the IEEE 1588 Hardware Assist driver is performed from a client driver communicating through the published API set described in the Intel[®] EP80579 Software Drivers for Embedded Applications Programmer's Guide and API Reference Manual. Intel leaves the design and development of a client driver to the customer since it is implementation specific.

5.6.4 Linux Sample Codelet

A codelet is provided to demonstrate how a sample application interfaces with the IEEE 1588 Hardware Assist driver. This codelet is intended to run on an Intel[®] EP80579 Integrated Processor with Intel[®] QuickAssist Technology Development Board with CentOS installed. The EP80579 Embedded Software kernel should also be installed, but the codelet works even if it is not. Run the codelet as follows:

1. Login as root. Root permissions are required for all operations.



2. Open a terminal window, change directory to *\$ICP_ROOT/Embedded/codelet/1588* and execute the bash script. Do not move the script to another location.

[root@localhost ~]# ./install.bash

The script checks for the IEEE 1588 Hardware Assist driver and client drivers, runs all of the makefiles in the correct order, loads the client driver, and runs the application.

The application output can be viewed in the terminal window.

3. Client driver output can be viewed in a second terminal window by printing /var/log/messages as follows:

[root@localhost ~]# tail -f /var/log/messages

4. For information about the 1588 codelet, refer to the Readme document located at \$ICP_ROOT/Embedded/codelet/1588.

5.7 Global Configuration Unit and Gigabit Ethernet Driver

Two drivers complete the software features set for network connectivity on the EP80579 integrated Gigabit Ethernet controllers; the Global Configuration Unit (GCU) driver and the Gigabit Ethernet (GbE) driver. The GCU driver controls the MAC and administrative activities. The GbE driver controls the network connectivity. The GbE driver is dependent on the GCU driver.

Note: The Global Configuration Unit driver must be installed prior to installation of the Gigabit Ethernet driver.

5.7.1 Linux Compilation Instructions

All source files for the Linux release of the Global Configuration Unit (GCU) driver and the Gigabit Ethernet (GbE) driver are located in the following directory within the Linux compatible EP80579 embedded software drivers release:

\$ICP ROOT/Embedded/src/GbE

Compilation of both the GCU and the GbE drivers, separately from the rest of the software package, is possible. Enter the *\$ICP_ROOT/Embedded/src/GbE* directory and execute the following commands:

make clean make

The Global Configuration Unit driver and the Gigabit Ethernet driver compiles and the resulting gcu.ko and iegbe.ko files are placed in the *\$ICP_ROOT/Embedded/src/GbE/build/linux_2.6/kernel_space* directory.

5.7.2 Linux Module Load/Unload Instructions

To load the Linux Global Configuration Unit driver and the Gigabit Ethernet driver, execute the following commands from the directory where the compiled executable resides (*\$ICP_ROOT/Embedded/src/GbE/build/linux_2.6/kernel_space*):

```
insmod gcu.ko
insmod iegbe.ko
```

To unload the Linux Global Configuration Unit driver and Gigabit Ethernet driver execute the following commands:

rmmod iegbe.ko



rmmod gcu.ko

The Ismod command may be used to confirm if a module has been loaded or unloaded:

lsmod | grep gcu lsmod | grep iegbe

The output of these Ismod command lists modules loaded in the system containing "gcu" and "iegbe" as part of their name.

The Global Configuration Unit driver and the Gigabit Ethernet driver can also be installed for persistency using the "make install" command from the *\$ICP_ROOT/Embedded/src/GbE* directory.

5.7.3 Linux Runtime Configuration of GCU and Gigabit Ethernet

Runtime configuration of network support is through traditional ifconfig commands, detailed are provided in man pages in open source.

5.8 SMBus

5.8.1 Linux Compilation Instructions

All source files for the SMBus Linux support are available in the Linux kernel's open source. Additionally, the Im_sensors rpm must be installed to work with additional SMBus hardware on the system board. The Im_sensors rpm can be acquired through the Red Hat Network (or other online sources).

Download the Im_sensors-2.10.0-3.1.i386 RPM package and install the rpm with the following command:

rpm -ivh lm_sensors-2.10.0-3.1.i386

Note: If Im_sensors package was previously installed, the above installation will return warning messages indicating package was previously installed.

The SMBus driver is a hardware access driver to the EP80579 integrated processor component.

Two additional SMBus drivers, which are available in open source, need to be installed prior to use of the i2c-i801 driver.

i2c_core i2c_dev

These drivers are found in open source at:

```
/drivers/i2c/i2c-core
/drivers/i2c/i2c-dev
```

It is likely that these drivers will be installed during installation of the operating system and kernel. To determine if they are installed perform an Ismod similar to the examples above.

5.8.2 Linux Module Load/Unload Instructions

If the additional SMBus driver files are not installed, install them with the modprobe command:

modprobe i2c_core



mobprobe i2c dev

The Ismod command may be used to confirm the modules have been loaded:

lsmod | grep i2c_core
lsmod | grep i2c dev

Installing the i2c-i801 driver is accomplished using the following command:

modprobe i2c i801

To unload the Linux SMBus i2c-i801 driver execute the following command:

rmmod i2c i801

The Ismod command may be used to confirm if a module has been loaded or unloaded:

lsmod | grep i2c i801

The output of this Ismod command lists modules loaded in the system containing "i2c_i801" as part of their name.

5.8.3 Linux Runtime Configuration of SMBus

Runtime configuration of the SMBus driver is performed from a client driver communicating through the published API set described in the Intel[®] EP80579 Software Drivers for Embedded Applications Programmer's Guide and API Reference Manual. Examples of already developed client drivers in open source are i2cdetect or i2cdump. Intel leaves the design and development of a client driver to the customer since it is implementation specific.



6.0 Booting from CompactFlash*

This chapter describes how to setup and boot the development board from a CompactFlash* (CF) card. Instructions are provided for legacy boot and EFI.

During the installation process, the following tasks will be performed:

- Install CentOS to CompactFlash card
- Create initial Ramdisk image

6.1 System Requirements

Before you begin, you must ensure the following system requirements are met:

- CompactFlash Memory Card Reader/Writer with USB interface
- CompactFlash card. 2GB or more is recommended to allow space for the CentOS image.
- Clean installation of CentOS* 5.2 on hard drive
- CentOS 5.2 installation DVD/CDs media
- Memory stick installed on development board
 - **Note:** For legacy boots, both 1 GB and 512 KB memory sticks are supported. For EFI boots, 512 KB memory stick is supported (1 GB is **not** supported). Refer to the Embedded Software release notes for additional information.
- SATA or USB DVD/CD-ROM connected to development board
- Network connection to internet (required to obtain elilo if using EFI boot)

6.2 Procedure

Pre-boot Firmware

- 1. Verify BIOS 64 is installed on the development board. The Project Version in the Main BIOS settings should have the value TRXTG 64.00.
- 2. Verify SATA mode is set to AHCI mode in the BIOS. This setting is available on the Advanced tab under IDE Configuration.
- 3. Update BIOS settings to boot from DVD/CD ROM drive. The setting is available on the Boot tab.
- 4. Shutdown the development board.

Install CentOS 5.2 from DVD/CD Media onto CompactFlash card

- 5. Plug in the CompactFlash Memory Card Reader/Writer and connect this to USB port on the development board.
- 6. Disconnect the SATA hard drive from the development board.
- 7. Place the CentOS DVD or CD #1 in your DVD/CD-ROM drive and boot the development board from the DVD/CD-ROM.



- 8. Begin installing CentOS onto CompactFlash media.
- 9. When the message "Installation requires partitioning of your drive." is displayed, select "Remove all partitions on selected drives and create default layout."
- 10. Click the "Review and modify partitioning layout" checkbox.
- 11. Click "Next"
- 12. Remove Logical Volume Manager partitions. This can be done by: Select the VolGroup00 and click Delete button.
 - a. Select /dev/sda2 LVM PV type partition and click Delete button.
 - b. Repeat for any additional LVM partitions on the drive.

Complete these steps (13. through 14.) for EFI Boot

- 13. Remove the /boot partition. This can be done by: Select /boot partition and click the Delete button.
- 14. Create "/" and "/boot/efi" partitions. "/boot/efi" partition must have format type set to "vfat". The CentOS CompactFlash drive partition sould look simular to:

/dev/sda1	1	ext3	1866
/dev/sda2	/boot/efi	vfat	101

Note: These partition sizes assume a 2 GB CompactFlash card.

Complete these steps (15. through 16.) for Legacy Boot

15. Create "/" partition. This can be done by:

- a. Select the free space created by deletion of the LVM partition(s).
- b. Click New.

The mount point should be "/", the file system type should be left at "ext3" and then select "Fill to maximum available size".

16. Leave the /boot partition as ext3

Complete these steps (17. through 20.) for both Legacy and EFI Boot

- 17. Click Next
 - *Note:* You will receive a warning notice that a swap partition was not specified. This warning can be ignored. Select "Yes" to continue with your requested partitioning.
- 18. Select the bootloader when prompted.
 - a. For Legacy Boot, select "The Grub boot loader will be installed"
 - b. For EFI Boot, select "No boot loader will be installed"
- 19. Use a minimal set of packages it is likely a KDE or GNOME desktop environment will not fit onto CompactFlash card. It is highly recommended to deselect any additional packages from the install.
- 20. After installation to the CompactFlash card, reconnect the hard drive and boot CentOS from the hard drive.

Build CompactFlash Driver

Note: These steps can be skipped if the top level make and make install commands were executed in Section 4.2, "Build All Embedded Software Drivers" on page 20.

21. Environmental Setup

```
$ export ICP_ENV_DIR=/EP805XX_release/Embedded
```

```
$ export ICP_ROOT=/EP805XX_release
```



22. Build CompactFlash Driver

```
$ cd /EP805XX_release/Embedded/src/CF
$ make
```

23. Copy CompactFlash into Modules.

```
$ mkdir -p /lib/modules/2.6.18-92.el5/kernel/drivers/CF
$ cp /EP805XX_release/Embedded/src/CF/build/linux_2.6/kernel_space/leb_cf.ko /lib/
modules/2.6.18-92.el5/kernel/drivers/CF/
```

Create Initial Ramdisk

24. Make the initial ramdisk with CompactFlash driver.

```
$ cd /EP805XX_release
$ mkinitrd --preload=leb_cf initrd-2.6.18-cf.img 2.6.18-92.el5
```

Populate CompactFlash

25. Plug in CompactFlash Memory Card Reader/Write adapter and connect to development board.

Complete these steps (26. through 30.) for EFI Boot

- 26. Download elilo-3.6-ia32.efi from: http://sourceforge.net/project/ showfiles.php?group_id=91879&package_id=97044 Expand the 3.6 Release to locate file. This download can be performed on the development board.
- 27. Rename the file to elilo.efi
- 28. Copy elilo.efi onto /boot/efi (vfat) partition of CompactFlash card.
- 29. Copy /EP805XX_release/initrd-2.6.18-cf.img onto /boot/efi (vfat) partition of CompactFlash card.
- 30. Create elilo.conf file on /boot/efi (vfat) partition of CompactFlash.

```
verbose=5
default=linux
image=vmlinuz-2.6.18-92.el5
label=linux
initrd=initrd-2.6.18-cf.img
read-only
root=/dev/cfal
```

Note: Root is set to /dev/cfa1 because during installation of CentOS 5.2 to CompactFlash the first partition was formatted as /.

Complete these steps (31. through 32.) for Legacy Boot

- 31. Copy /EP805XX_release/initrd-2.6.18-cf.img onto /boot partition of CompactFlash card.
- 32. Edit /boot/grub.conf on /boot/grub partition of CompactFlash and replace "initrd/initrd-2.6.18-92.el5.img" with "initrd/initrd-2.6.18-cf.img"

Boot from CompactFlash

- 33. Shut down the development board and disconnect SATA hard drive from development platform.
- 34. Plug the CompactFlash card into development board (not the CompactFlash Memory Card Reader/Write adapter).
- 35. Modify SATA mode to Legacy mode in the BIOS. This setting is available on the Advanced tab under IDE Configuration.



For Legacy Boot the required work is complete. Booting the system now will begin loading CentOS from CompactFlash. Instructions for EFI Boot are contained below.

Complete these steps (36. through 38.) for EFI Boot

- 36. Boot to system to the EFI shell. You may need to update the Boot order in BIOS to select EFI shell as first boot device. The setting is located on the Boot tab in the BIOS settings.
- 37. Switch to fs0.

Shell> fs0:

38. From EFI shell, run elilo.

fs0:\> elilo

This command initiates the CentOS boot.



7.0 Pre-boot (BIOS) Firmware

The pre-boot firmware is executed when the system is powered up or reset. It initializes and configures system memory, devices and buses/interfaces.

The pre-boot firmware is based on the AMI Aptio* 4.5 core and compliant to EFI v1.1. The firmware is stored in the Firmware Hub (FWH) or SPI (Serial Peripheral Interface) Flash; the FWH or SPI Flash can be updated using a flash utility tool that is provided by Intel or by using a floppy drive connected to the floppy header.

The pre-boot firmware setup menu can be used to view and modify the system settings for the development board. The setup menu is accessed by pressing the key during pre-boot firmware boot up (before the operating system boot begins). The setup menu bar is shown in Table 2.

7.1 Pre-boot Firmware Setup Menu

Table 2 shows the pre-boot firmware setup main menu and provides a brief description of each menu option. Table 3 shows the function keys that can be used when navigating and selecting options from pre-boot firmware menus.

Table 2.Pre-boot Firmware Setup Main Menu

Main	Advanced	Chipset	Security	Boot	Exit
Displays processor and memory configuration Setup for CMOS system date and time	Configures advanced features and settings	Configures different major components	Setup passwords and security features	Selects boot options and configurations	Saves or discards changes to setup program options

Table 3. Pre-boot Firmware Setup Program Function Keys

Function Key	Description
< or >	Moves cursor left or right in the main menu
^ or v	Moves cursor up or down to select sub-menu items
Enter	Executes command or selects the submenu
F7	Discard changes
F8	Load the fail-safe default
F9	Load the optimal default configuration for the current menu
F10	Save the current configuration and exit the setup menu
ESC	Exit the setup menu



7.1.1 Serial Console Redirection

The pre-boot firmware supports redirection of both video and keyboard via a serial port. When console redirection is enabled, the remote console terminal sends keystrokes to the Intel[®] EP80579 Development Board pre-boot firmware and the pre-boot firmware redirects the video to the console terminal.

As an option, the Intel[®] EP80579 Development Board can be operated without keyboard or video and can run entirely via the remote serial console. This includes accessing the pre-boot firmware setup menu.

Console redirection ends when operating system boot up begins. After boot up begins, the operating system is responsible for continuing the redirection.

Note: pre-boot firmware console redirection is text only. Graphical data, such as logos, are not redirected.

Table 4 shows the default settings of the serial console redirection.

Table 4. Serial Console Redirection Default Settings

Parameter	Default
Port Number	COM 1
Baud Rate	115200
Data Bits	8
Parity	None
Stop bits	1
Flow Control	None

7.1.2 Changing the Boot Device

Use the following procedure to change the boot device:

- 1. Press the key during POST to enter the pre-boot firmware setup menu.
- 2. Use the arrow keys to navigate to the <BOOT> menu.
- 3. Move the cursor to <Boot Device Priority>.
- 4. Select the desired booting sequence list.
- *Note:* Follow the instructions on the right side of the pre-boot firmware screen to navigate and change pre-boot firmware settings.

7.1.3 Maximum Memory Speed Setup

The maximum memory speed supported on the development board can be selected using the Maximum Memory Speed Setup option available in the BIOS Setup Menu on the Chipset tab.

Enter the BIOS Setup Menu and select the Chipset tab. Select North bridge, and navigate down to the bottom option, titled Max Memory Speed Support. Select this option using the Enter button. A selection box appears providing the following options:

- 400 MHz
- 533 MHz
- 667 MHz
- 800 MHz



The default setting in the BIOS is 400 MHz. If a higher speed memory DIMM is inserted into the development board, the corresponding memory speed must be selected in the BIOS Setup Menu to support the intended speed. Otherwise, the memory is reduced to the default of 400 MHz.

7.1.4 Coherent and Non-Coherent Memory Allocation

The development board supports allocation of memory regions for coherent and noncoherent use. Coherent and non-coherent memory use features are for development boards that use the Intel[®] EP80579 Integrated Processor with Intel[®] QuickAssist Technology. Intel[®] EP80579 Integrated Processors (without Intel[®] QuickAssist Technology) do not make use of the memory set aside for these features. Therefore, it is recommended that the settings for coherent and non-coherent usage be set to zero so that no memory is allocated to these features.

Enter the BIOS Setup Menu and select the Chipset tab. Select North bridge, and navigate down towards the bottom to the Coherent Mem Size option and press Enter to select this option. A dialog box is displayed prompting the user to enter a value. Type the numerical value zero, "0", and press Enter. Navigate to the next option, Non-Coherent Mem Size, and press Enter to select this option. A dialog box is displayed prompting the user to enter a value. Again, type the numerical value zero, "0", and press enter. These selections override the BIOS default settings and allocate no memory regions for these two features.

Note: This software package requires the pre-boot firmware (BIOS) for your hardware to allocate the values for each region called out in Table 5. For more information on these regions, refer to the Intel[®] EP80579 Integrated Processor Product Line Datasheet, Section 3.0.

Table 5.Memory Allocation Settings

Datasheet name	Software name	Region Size
IA/ASU Shared (Coherent)	CDRAM	0
IA/ASU Shared (AIOC-Direct)	NCDRAM	0

7.1.5 Legacy and AHCI SATA Mode

The development board supports hard drives in legacy SATA mode and in Advanced Host Controller Interface (AHCI) mode. AHCI mode provides advanced capabilities and improved performance, provided the hard drive supports the following features:

- Hot Plug
- Native Command Queuing
- Speeds up to 3 Gb/s

Refer to the Serial ATA Organization web site for more information:

http://www.serialata.org/

The development board pre-boot firmware (BIOS) can be configured in either Legacy or AHCI mode as desired. The BIOS defaults to Legacy mode because not all hard drives support AHCI. To toggle the BIOS to either Legacy or AHCI mode, proceed as follows:

- 1. Press the key during POST to enter the pre-boot firmware setup menu.
- 2. Use the arrow keys to navigate to the Advanced menu.



- 3. Use the arrow keys to navigate to the IDE Configuration option.
- 4. Select the IDE Configuration option.
- 5. Use the arrow keys to navigate to the SATA Mode option.
- 6. Press the Enter key. A SATA Mode popup window appears.
- 7. Select either Legacy or AHCI as desired. Do not use Native as a selection.
- 8. Press F4 to save.
- 9. Choose Yes. The system continues the boot process.

7.2 Pre-boot Firmware Image Reflashing Instructions

One method is available for updating the pre-boot firmware flash images located on the development board firmware hub:

• AFUEFI Flash Recovery

It is possible that updated pre-boot firmware images will become available from Intel for the Intel[®] EP80579 Integrated Processor with Intel[®] QuickAssist Technology Development Board. The latest pre-boot firmware image is available from Intel's public web site, http://www.intel.com/go/soc located with all other collateral related to the EP80579 integrated processor.

If the pre-boot firmware image should become corrupted on the board, also utilize these instructions to reflash the image to the firmware hub.

7.2.1 Aptio Flash Update Utility (AFUEFI)

Use the following instructions to update the development board pre-boot firmware image using a USB memory stick and the Aptio Flash Update Utility from AMI.

Necessary Hardware:

- · development board
- Socketed Firmware Hub
- USB Memory Stick

Necessary Software:

- · development board pre-boot firmware Image
- Aptio Flash Update Utility AFUEFI

Steps to Reflash Image:

- 1. Load the AFUEFI utility onto the USB memory stick.
- 2. Load the pre-boot firmware image onto the USB memory stick.
- 3. Boot the development board to the EFI shell. Change the boot setting in the BIOS Setup Selection to boot from the EFI shell if needed.
- 4. Insert the USB memory stick into the USB port.
- 5. Once the USB memory stick is recognized on the system (activity seen on the USB memory stick), several commands are available as follows:
 - Type "map -r" to list all devices available.
 - Type "fs0:" to enter USB device.
 - Type "Is" to list all files.



- 6. Once the "fs0:" command has been initiated, execute the AFUEFI utility.
 - Type "AFUEFI <pre-boot firmware image name> /X /P /B /N"
 - (the <pre-boot firmware image name> will be similar to TRXTG055.ROM)
- 7. Reboot the development board once reflashing has completed.
- 8. Confirm the image has been updated to the reflashed image by looking in BIOS Setup.



8.0 Uninstalling the Software

Please refer to the instructions for individual loading and unloading modules in Chapter 5.0, "Runtime Configuration". To uninstall all drivers, see the last paragraph in Section 4.3, "Uninstalling Embedded Software Drivers" on page 21.

8.1 Linux Module/Driver Dependencies

Table 6 lists the dependencies for the driver modules or patch within the Intel[®] EP80579 Software Drivers for Embedded Applications package. OS installation is assumed.

Table 6. Linux Module/Driver Dependencies

Module/Driver/Patch	Dependency 1	Dependency 2
EP80579 integrated processor Linux Patch	OS Source	None
Controller Area Network (CAN)	None	-
Enhanced DMA (EDMA)	None	-
CompactFlash (CF)	None	-
IEEE 1588 Hardware Assist (1588)	None	-
General Purpose IO (GPIO)	None	-
Gigabit Ethernet (GbE)	GCU	None
Global Configuration Unit (GCU)	None	-
SMBus (i2c-i801)	i2c-core	i2c-dev
Watchdog Timer (WDT)	None	-



9.0 Troubleshooting

- It is advised to plug the Matrox Millennium G550 PCIe* graphics card into the PCI Express* slot closest to memory. Lack of video has been exhibited in the PCI Express x4 slot.
- When using the Matrox Millennium G550 PCIe graphics card, utilize the bottom video port on this dual port video card. The top port will not output any video to the display.
- Some LCD monitors will not function properly with the Intel[®] EP80579 Integrated Processor with Intel[®] QuickAssist Technology Development Board. In one case, an Acer monitor, model AL1916 C reported "Input Not Supported" upon boot complete. Try an alternative LCD monitor if no video support is displayed.



10.0 Glossary

AHCI	Advanced Host Controller Interface
CAN	Controller Area Network
EDMA	Enhanced Direct Memory Access
GbE	Gigabit Ethernet
GCU	Global Configuration Unit
GPIO	General Purpose Input Output
IEEE	Institute of Electrical and Electronics Engineers
IHS	Integrated Heat Spreader
SMbus	System Management Bus
TIM	Thermal Interface Material
WDT	Watchdog Timer

§§