

ZeBu[®] Server

Installation Manual

Version O-2018.09-2, March 2019



Copyright Notice and Proprietary Information

©2019 Synopsys, Inc. All rights reserved. This Synopsys software and all associated documentation are proprietary to Synopsys, Inc. and may only be used pursuant to the terms and conditions of a written license agreement with Synopsys, Inc. All other use, reproduction, modification, or distribution of the Synopsys software or the associated documentation is strictly prohibited.

Destination Control Statement

All technical data contained in this publication is subject to the export control laws of the United States of America. Disclosure to nationals of other countries contrary to United States law is prohibited. It is the reader's responsibility to determine the applicable regulations and to comply with them.

Disclaimer

SYNOPSYS, INC., AND ITS LICENSORS MAKE NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

Trademarks

Synopsys and certain Synopsys product names are trademarks of Synopsys, as set forth at <http://www.synopsys.com/company/legal/trademarks-brands.html>. All other product or company names may be trademarks of their respective owners.

Free and Open-Source Software Licensing Notices

If applicable, Free and Open-Source Software (FOSS) licensing notices are available in the product installation.

Third-Party Links

Any links to third-party websites included in this document are for your convenience only. Synopsys does not endorse and is not responsible for such websites and their practices, including privacy practices, availability, and content.

www.synopsys.com

Safety Recommendations

The ZeBu Server unit must be used, installed, and stored as described in this manual. The ZeBu Server unit is intended for indoor use only. Before connecting the ZeBu Server unit to your PC(s), hard core(s) or target(s), the power to the PC must be switched OFF, and the power cord removed. Internal voltages inside the PC are potentially lethal. The ZeBu Server unit is not intended to be serviced on-site. Service operations should only be performed by Synopsys qualified and trained personnel/representatives. It is the user's responsibility to ensure that the ZeBu Server unit is used in safe working conditions. Do not remove covers or leave them off while in use.

In the event of damage of the ZeBu Server unit due to an action or omission by the user (including the actions or omissions described hereafter), the user shall promptly notify Synopsys and shall follow Synopsys's instructions based on your License Agreement.

Damaged ZeBu Server unit replacement limitations:

Synopsys shall not replace the ZeBu Server unit in case of improper use or of damage caused by the user, including the following cases:

- Damage during shipment other than shipment from Synopsys to the user;
- Damage caused by accidents such as objects dropping, board falling to the ground, spilled liquid, or immersion in liquid;
- Damage caused by failure to provide a suitable installation environment for the ZeBu Server unit, as described in this manual;
- Damage caused by installation of the ZeBu Server unit that is not compliant with this manual;
- Damage caused by an overheat due to the use of the ZeBu Server unit in a room exceeding the normal operating temperature range indicated in this manual;
- Damage caused by power failure or power surges;
- Damage caused by use of the ZeBu Server unit for purposes or in any manner other than those indicated in this manual;
- Damage caused by short-circuit coming from a target system or produced by the user on an in circuit emulation connector;
- ZeBu Server unit configuration has been modified by the user;

- Any minor ZeBu Server unit malfunction that would not alter the functionalities or that would not significantly impact the level of performance of the Product as agreed between the Parties;
- Damage caused by conditions of use or storage of the ZeBu Server unit which do not conform to this manual.

Any modification of the configuration of the ZeBu Server unit or installation of an accessory or upgrade shall be performed by Synopsys. No maintenance services, update or damaged ZeBu Server unit replacement will be performed on ZeBu Server unit or on ZeBu Server unit's configuration altered by the user.

Any accessory or upgrade shall remain Synopsys's property, and upon termination of the agreement executed between Synopsys and the user, the user shall return all Synopsys property to Synopsys. For technical assistance, contact zebu_support@synopsys.com

Contents

1. Safety Recommendations	iii
1. Preface	15
About This Book	15
Intended Audience	15
Contents of This Book	15
1. ZeBu Server Hardware	17
1.1. Overview	18
1.1.1. Design Capacity	23
1.2. ZeBu Server Hardware for Server 1, Server 2, and Server 3	25
1.2.1. 2-slot ZeBu Server Unit	25
1.2.2. 5-slot ZeBu Server Unit	26
1.3. ZeBu Server Hardware for Server 4	28
1.4. Interconnection With Host PCs and Other Units	28
1.4.1. For ZeBu Server 1, 2, and 3	29
1.4.2. For ZeBu Server 4	29
1.5. Single Unit System Cabling	30
1.6. Multi Unit Interconnection Topology	30
1.7. Multi User Configurations for ZeBu Server 1, 2, and 3	31
1.8. Multi User Configurations for ZeBu Server 4	32
1.9. Multi PCIe Cabling for ZeBu Server 1, 2, and 3	32
1.10. Multi PCIe Cabling for ZeBu Server 4	32
2. PC Requirements for Installation	33
2.1. Hardware Requirements	34
2.1.1. Physical Connection in the PC	34
2.2. Software Requirements	39
2.2.1. Linux Operating System Requirements	39

3. Hardware Installation	45
3.1. Installing the ZeBu Server PCIe Board in a Host PC.....	46
3.2. Removing the ZeBu Server PCIe Board from a Host PC.....	47
3.3. Connecting the ZeBu Server PCIe Cables to the Host PC	48
3.4. Connecting ZeBu Server Units to the Power Supply.....	49
3.5. Changing the Power Supply Fuse on a ZeBu Server Unit.....	50
3.6. Getting Board Labels	51
4. Software Installation	53
4.1. Read This First	54
4.2. Downloading the Packages for Installation	55
4.2.1. Downloading and extracting the Synopsys Installer	55
4.2.2. Downloading the ZeBu Software Package.....	56
4.3. Installing the ZeBu Software.....	58
4.3.1. In Graphical Mode	58
4.3.2. In Script Mode	60
4.4. Diagnostics Patches Overview.....	76
4.4.1. Diagnostics Packages	76
4.4.2. Diagnostics Patches Description for ZeBu Server 1	78
4.4.3. Diagnostics Patches Description for ZeBu Server 2	80
4.4.4. Diagnostics Patches Description for ZeBu Server 3	82
4.5. Diagnostics Patches Description for ZeBu Server 4.....	84
4.5.1. Example Unit Configuration of ZeBu Server 4	84
4.6. Post- Installation	84
4.7. Installing Licenses for ZeBu	85
4.7.1. Installation Process for Licenses	85
4.7.2. Modifying the License File	86
4.7.3. Starting the License Server.....	89
4.7.4. Updating the ZeBu License File.....	91
4.8. Downloading Required Interoperable Technologies	91
4.8.1. Unified Compile Feature.....	91
4.8.2. Debugging Features	91
4.8.3. Co-Simulation with VCS.....	91
5. Configuring End-User's Environment.....	93
5.1. Mandatory Environment Variables for ZeBu Software.....	94
5.1.1. ZeBu Compilation and Runtime Software	94

5.1.2. Enhanced Compilation Feature for ZeBu Server 2 Hardware Only	94
5.2. Activating a User License	95
5.2.1. Activating a ZeBu License	95
5.2.2. Activating a Xilinx License.....	95
5.3. Declaring the Setup Directory for Emulation Runtime	96
5.4. Runtime Settings for Relocation.....	96
5.4.1. Runtime Settings for Relocation With a Multi Unit/Multi PCIe System .	98
5.4.2. Runtime Settings for Relocation on ZeBu Server 4	99
 6. Initializing the Host PCs	 101
6.1. Prerequisites.....	102
6.2. Initializing PCIe Boards on Host PC Running ZeBu 2017.03 or Newer Releases.....	103
6.2.1. Installing Driver on Host PC	103
6.2.2. Removing Driver From Host PC	104
6.3. Initializing PCIe Boards on Host PC Running ZeBu 2017.03 and Older Releases (2015.03 or newer)	104
6.3.1. Preparing the PC for ZeBu Releases Installation	105
6.3.2. Initializing the PCIe Boards for ZeBu 2017.03 and Newer Releases..	105
6.3.3. Initializing the PCIe Boards for ZeBu Releases Older Than 2017.03 (2015.03 or newer)	106
6.4. Initializing the PCIe Boards Automatically When Booting Linux....	107
6.4.1. Pre-compiling the ZeBu Device Driver on Host PC	107
6.5. PCIe Boards Detection Order.....	108
6.6. zInstall Log Files	111
6.6.1. Connection/Disconnection Logs	111
6.7. Updating PCIe Boards Firmware	113
6.7.1. Updating Only One PCIe Board in Host PC.....	113
6.7.2. Updating Several PCIe Boards in Host PC.....	113
 7. ZeBu Server System Setup	 115
7.1. zSetupSystem Tool.....	116
7.2. Choosing the Setup Directory.....	117
7.2.1. One Directory for Each System.....	117
7.3. Preparing the Setup File.....	118
7.3.1. Setup File Template	118
7.3.2. Declaring the Setup Directory in the zini file.....	119

7.3.3. Declaring the diagnostics patches (if several are installed)	119
7.3.4. Managing Calibration History	119
7.3.5. Memory Tests.....	120
7.3.6. Reserving Memory for SRAM-Trace (Static-Probes)	124
7.4. Launching zSetupSystem.....	128
7.4.1. Multiple Host PCs.....	129
7.4.2. Creating the Host Topology	129
7.5. Content of the Setup Directory	131
7.5.1. Setup Directory Files	131
7.5.2. Sub-Directories in the Setup Directory	131
7.5.3. Information Log Files	132
7.5.4. ZeBu Runtime Log Files	139
8. Initializing the ZeBu Server System	143
8.1. Initializing the System With zUtils -initSystem.....	144
9. Target Hardware Configuration File	145
9.1. Generating a Configuration File for One ZeBu Server System	147
9.2. Generating a Configuration File for Several ZeBu Server Systems	147
9.2.1. Hardware Configuration File for Identical ZeBu Server Systems.....	147
9.2.2. Hardware Configuration File for Non-Identical ZeBu Server Systems	148
9.3. Generating a Hardware Configuration File for Relocation	149
9.4. Generating a Default Configuration File.....	149
9.4.1. Default Configuration File Using a List of Modules	150
9.4.2. Default Configuration File Using an FPGA Count	154
9.4.3. Default Configuration File for a Specific Interconnection Topology ...	154
9.5. Using a Sample Configuration File	156
9.5.1. ZeBu Server 1 Sample Configuration Files	156
9.5.2. ZeBu Server 2 Sample Configuration Files	158
9.5.3. ZeBu Server 3 Sample Configuration Files	160
9.5.4. ZeBu Server 4 Sample Configuration Files	161
10. Troubleshooting	163
10.1. Missing Kernel Source Files for the Linux Kernel	164
10.2. ZeBu Server Unit not Detected by The PC	166
10.3. ZeBu Kernel Module Version Mismatch	166
10.4. ZeBu Software Does Not Recognize the Hardware.....	167

10.5. zInstall Accidentally Called	167
10.5.1. Problem Description	167
10.5.2. Solution	168
10.6. zServer or zUtils is Running While Trying to Install a Driver	168
10.6.1. Problem Description	168
10.6.2. Solution	168
10.7. ZeBu Driver Not Installed (Versions prior to L-2016.06)	169
10.7.1. Problem Description	169
10.7.2. Solution	169
10.8. Problems When Compiling With gcc	170
10.9. Cannot Communicate With Units When a PC is OFF	171
10.10. Runtime Errors Caused by Faulty Memories	172
10.10.1. Faulty DDR2 Memory for the SRAM Trace Memory	172
10.10.2. Faulty DDR2 and RLDRAM Memories	172
10.10.3. Faulty Mx_FS Memory	172
10.10.4. Faulty DDR3 Memories	173
10.10.5. Faulty Mx_FS DDR3 Memory	174
10.11. Synopsys Installer Does not Install Packages	175
10.11.1. Problem Description	175
10.11.2. Solution	176
 11. Appendix A: Technical Data	 177
11.1. Size and Weight	178
11.1.1. ZeBu Server 1, ZeBu Server 2, and ZeBu Server 3	178
11.1.2. ZeBu Server 4	178
11.1.3. Racks and Cabinets	179
11.2. Environment	180
11.2.1. Uninterruptible Power Supply	180
11.2.2. Environmental Constraints	180
11.2.3. Automatic Temperature Control	181
11.3. Power Supply	182
11.3.1. For ZeBu Server 1	182
11.3.2. For ZeBu Server 2	183
11.3.3. For ZeBu Server 3	184
11.3.4. For ZeBu Server 4	185
11.4. PCIe Interconnection Board and Cable	185
11.4.1. ZeBu Server 1, ZeBu Server 2, and ZeBu Server 3	185
11.4.2. ZeBu Server 4	186

11.5. Power Cords	186
11.5.1. ZeBu Server 1, ZeBu Server 2, and ZeBu Server 3	186
11.5.2. ZeBu Server 4	187
12. Appendix B: ZeBu Device Driver Switching Solution.....	189
12.1. Overview	190
12.1.1. Package Description	190
12.1.2. Device Driver Switching Solution.....	191
12.1.3. Troubleshooting	194
13. Appendix C: setup_template.zini.....	195
13.1. ZeBu Server 1	196
13.2. ZeBu Server 2	201
13.3. ZeBu Server 3	206

List of Figures

Architecture of an FPGA Module in ZeBu Server 1	21
Architecture of an FPGA Module in ZeBu Server 2	21
Architecture of an FPGA Module in ZeBu Server 3	22
Architecture of an FPGA Module in ZeBu Server 4 (multi unit)	22
Front Panel of a 2-slot Unit	25
Rear Panel of a 2-slot Unit	26
: Front panel of a 5-slot unit.....	27
Rear Panel of a 5-slot Unit	28
Single-unit System Cabling (2- and 5-slot units).....	30
Device Driver Switching Description	190

List of Tables

FPGA Types and FPGA Module Types in ZeBu Server	19
Design Capacity per ZeBu Server 1 Module	23
Design Capacity per ZeBu Server 2 Module	24
Design Capacity per ZeBu Server 3 Module	24
Design Capacity per ZeBu Server 4 Module	24
Number of Users per ZeBu Server System.....	31
ZeBu Server Configuration	32
Memory Requirements for ZeBu Server 1, 2, and 3	35
Synopsys Interoperable Technologies.....	41
Tested Third-party Tools	42
ZeBu Server Fuse Information	50
Bitstream Cache Capacity (default SRAM-trace size = 256 MBytes)	126
Bitstream Cache Capacity (SRAM-trace size = 0)	126
Bitstream Cache Capacity (max. SRAM-trace size for 8C modules = 2 GBytes)	126
Bitstream Cache Capacity (max. SRAM-trace size for 4C/16C modules = 4 GBytes)	127
Bitstream Cache Capacity (default SRAM-trace size = 256 MBytes)	127
Bitstream Cache Capacity (SRAM-trace size = 0)	127
Bitstream Cache Capacity (SRAM-trace size = 2 GBytes)	127
Bitstream Cache Capacity for ZeBu Server 3	128
Bitstream Cache Capacity for ZeBu Server 4	128
Files in the Setup Directory	131
Subdirectories in the Setup Directory.....	131
ZeBu Server 1 Syntax Rules	150
ZeBu Server 3 Syntax Rules	151
ZeBu Server 4 Syntax Rules	151
ZeBu Server 2 Syntax Rules	151
Available Sample Configuration Files	156
Available Sample Configuration Files	158

Available Sample Configuration Files.....	160
Size and Weight for ZeBu Server 1, ZeBu Server 2, and ZeBu Server 3	178
Racks and Cabinet Recommendations	179
Environmental Constraints	180
Power Consumption for a System With up to 32 FPGAs.....	182
Power Consumption for a System With More Than 32 FPGAs	182
Voltage Range for the ZeBu Server 2 Unit	183
Maximum Power Consumption	183
Voltage Range for the ZeBu Server 3 Unit	184
Maximum Power Consumption	184
Dimensions of the PCIe Interconnection Board	185
Dimensions of the PCIe Interconnection Board	186
ZeBu Server Power Cables.....	187

About This Book

This manual provides details about the physical architecture of ZeBu Server 1, ZeBu Server 2, ZeBu Server 3, and ZeBu Server 4 systems, with one or several ZeBu Server units.

This manual contains information about hardware and software prerequisites for a correct installation and describes the necessary procedures for hardware and software installation steps that can be done by end-users.

This manual also includes troubleshooting information.

Note

Throughout this manual, the term “ZeBu Server system” refers to a single unit or multi unit configuration for ZeBu Server 1, ZeBu Server 2, ZeBu Server 3, and ZeBu Server 4.

Intended Audience

This manual is written for experienced EDA hardware and software engineers to help them installing their ZeBu Server unit. These engineers should have experience with the Linux operating system.

Contents of This Book

The ZeBu Server Installation Manual has the following chapters:

Chapter	Describes...
“ZeBu Server Hardware”	Architecture and technology of ZeBu Server hardware
“PC Requirements for Installation”	Hardware and software requirements for ZeBu installation
“Hardware Installation”	Procedure for installing the ZeBu hardware
“Software Installation”	Procedure for installing the ZeBu software

Chapter	Describes...
<i>"Configuring End-User's Environment"</i>	Operations for configuring for each ZeBu user
<i>"Initializing the Host PCs"</i>	Initialization of PCIe boards on host PC
<i>"ZeBu Server System Setup"</i>	zSetupSystem tool for completing the ZeBu Server setup process
<i>"Initializing the ZeBu Server System"</i>	Initialization of ZeBu Server with the zSetupSystem tool
<i>"Target Hardware Configuration File"</i>	Modules plugged in each unit of a ZeBu Server system
<i>"Troubleshooting"</i>	Troubleshooting scenarios and workaround for each scenario
<i>"Appendix A: Technical Data"</i>	Technical data for ZeBu Server hardware
<i>"Appendix B: ZeBu Device Driver Switching Solution"</i>	Usage of ZeBu Device Driver Switching Solution
<i>"Appendix C: setup_template.zini"</i>	setup_template.zini file for each ZeBu Server

1 ZeBu Server Hardware

This chapter provides information about the ZeBu Server hardware and consists of the following sub-sections:

- [*Overview*](#)
- [*ZeBu Server Hardware for Server 1, Server 2, and Server 3*](#)
- [*ZeBu Server Hardware for Server 4*](#)
- [*Interconnection With Host PCs and Other Units*](#)
- [*Single Unit System Cabling*](#)
- [*Multi Unit Interconnection Topology*](#)
- [*Multi User Configurations for ZeBu Server 1, 2, and 3*](#)
- [*Multi PCIe Cabling for ZeBu Server 1, 2, and 3*](#)

1.1 Overview

ZeBu Server is an extremely high-capacity system emulator with easy setup. It is available in either 2-slot or 5-slot units to handle designs from:

- 10M to 100M ASIC-equivalent gates for ZeBu Server 1
- 12M to 200M ASIC-equivalent gates for ZeBu Server 2
- 60M to 300M ASIC-equivalent gates for ZeBu Server 3
- 150M to 600M ASIC-equivalent gates for ZeBu Server 4

For ZeBu Server 1, 2 and 3, from 2 users for a 2-slot unit to 5 users for a 5-slot unit can connect at the same time.

For ZeBu Server 4, up-to 8 users for a 4-slot unit can connect at the same time.

ZeBu Server is also expandable in a multi unit environment to accommodate up to 49 users (for a 10-unit system) on ZS 1, ZS 2, and ZS 3 and 32 users (for a 32-unit system) on ZS 4 and up to:

- 1 billion ASIC-equivalent gates for ZeBu Server 1
- 2 billion ASIC-equivalent gates for ZeBu Server 2
- 3 billion ASIC-equivalent gates for ZeBu Server 3
- 9 billion ASIC-equivalent gates for ZeBu Server 4

ZeBu Server is a scalable system and different types of FPGA modules are available to map the Design Under Test (DUT), all using Xilinx Virtex FPGAs.

TABLE 1 FPGA Types and FPGA Module Types in ZeBu Server

ZeBu Server Type	FPGAs Type	FPGA Module	Description
ZeBu Server 1	Xilinx Virtex-5 LX330	4C	4 FPGAs to map design with 2 GBits of RLD RAM per FPGA and 1 FPGA with 4 GBytes of DDR2 memory 1 FPGA to map RTB
		8C/ICE	8 FPGAs to map design with 1 GBits of RLD RAM per FPGA and 1 FPGA with 2 GBytes of DDR2 memory 1 FPGA to map RTB Direct ICE interface (1,200 data pins and dedicated clock pins) to connect a target system.
		16C	16 FPGAs with no additional memory and 1 FPGA with 4 GBytes of DDR2 memory to map design 1 FPGA to map RTB
ZeBu Server 2	Xilinx Virtex-6 LX760	5F/ICE	4 FPGAs with 1 GBits of RLD RAM per FPGA and 1 FPGA with 4 GBytes of DDR2 memory to map the design 1 FPGA to map the RTB Direct ICE interface (1,200 data pins and dedicated clock pins) to connect a target system.
		9F/ICE	4 FPGAs with 4 GBits of DDR2 memory per FPGA, 1 FPGA with 4 GBytes of DDR2 memory and 4 FPGAs with no additional memory to map the design 1 FPGA to map the RTB Direct ICE interface (1,200 data pins and dedicated clock pins) to connect a target system.
		17F	16 FPGAs with no additional memory and 1 FPGA with 4 GBytes of DDR2 memory to map the design 1 FPGA to map the RTB

TABLE 1 FPGA Types and FPGA Module Types in ZeBu Server

ZeBu Server Type	FPGAs Type	FPGA Module	Description
ZeBu Server 3	Xilinx Virtex-7 LX2000	9F	4 FPGAs with 512MB DDR3 memory per FPGA to map the design 1 FPGA with 2x 8GB DDR3 memory to map the design 4 FPGAs with no additional memory to map the design 1 FPGA to map the RTB
		9F/ICE	4 FPGAs with 512MB DDR3 memory per FPGA to map the design 1 FPGA with 2x 8GB DDR3 memory to map the design 4 FPGAs with no additional memory to map the design 1 FPGA to map the RTB Direct ICE interface (568 data pins and dedicated clock pins) to connect a target system.
ZeBu Server 4	Xilinx Ultrascale XCVU440	12F	2 FPGAs with 8GB DDR3 memory to map the design 6 FPGAs with 2x1GB of DDR3 memory to map the design 4 FPGAs with no additional memory

Through this technology, ZeBu Server supports various software and hardware debugging modes that can undertake the most challenging verification problems presented by today's electronics products during the entire design cycle.

Overview

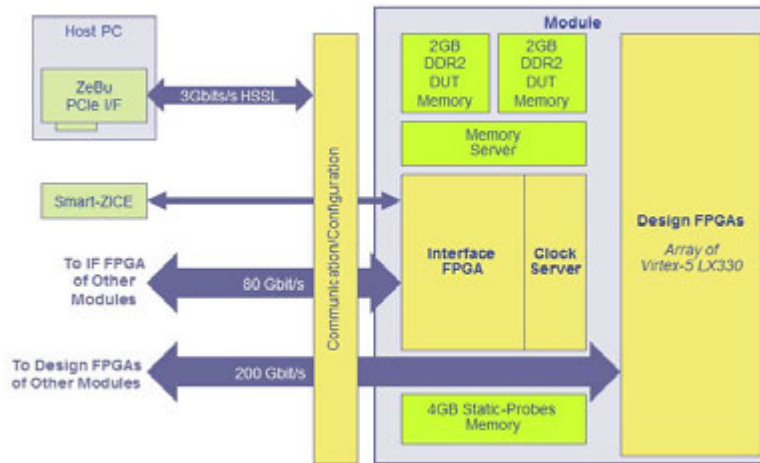


FIGURE 1. Architecture of an FPGA Module in ZeBu Server 1

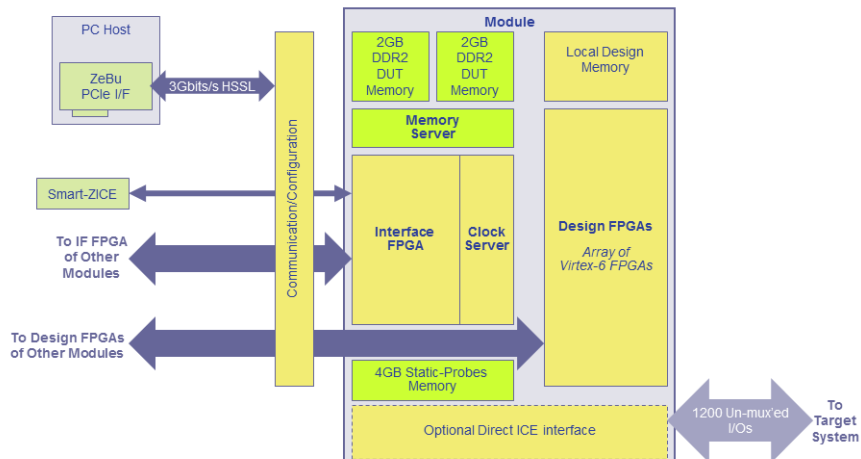


FIGURE 2. Architecture of an FPGA Module in ZeBu Server 2

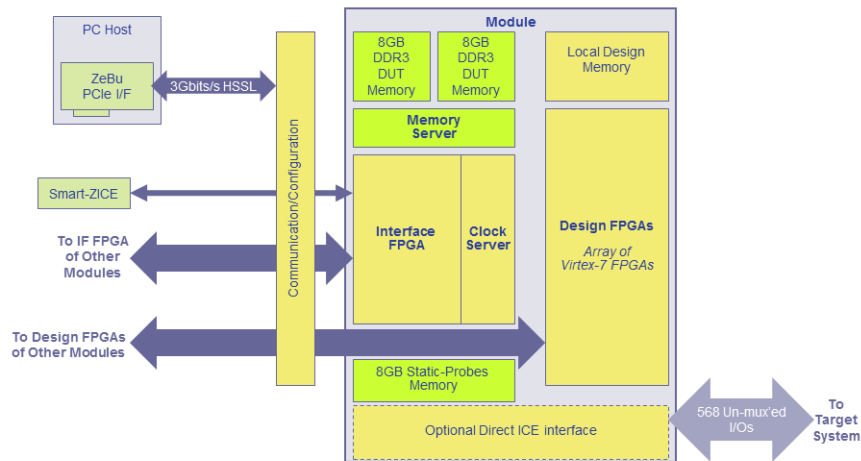


FIGURE 3. Architecture of an FPGA Module in ZeBu Server 3

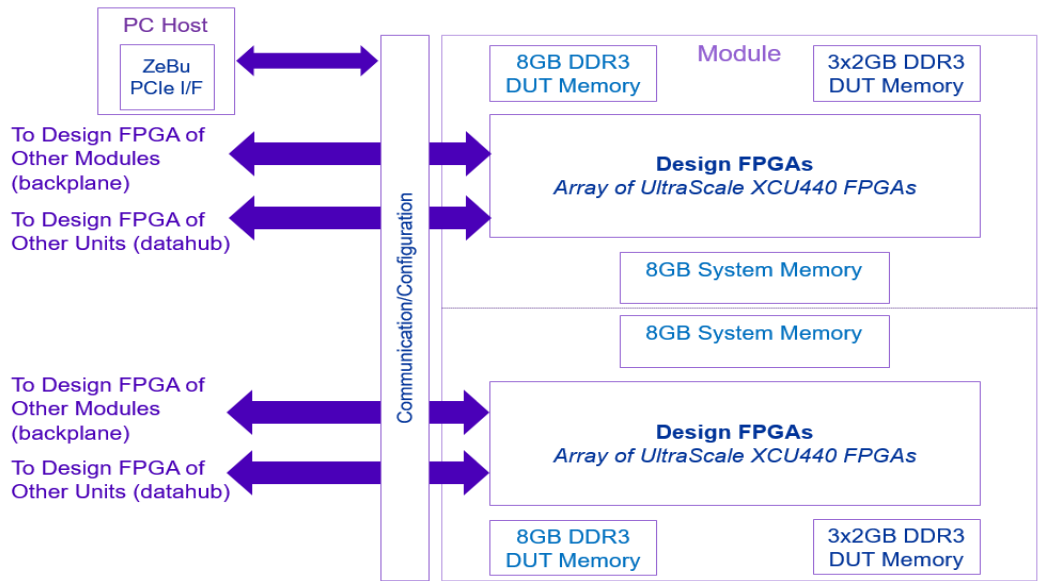


FIGURE 4. Architecture of an FPGA Module in ZeBu Server 4 (multi unit)

The ZeBu Server unit is connected to the host PC through a PCI Express interconnection board. In multi user environments, a different PC can be connected to each module of the unit.

ZeBu Server also provides two additional interfaces for connecting the DUT to a software debugger or a target system:

The Smart Z-ICE interface (64 data pins and 4 clock pins for a 2-slot unit; 80 data pins and 5 clock pins on a 5-slot unit) provides the support of standard software debuggers such as JTAG cables. This interface and its usage are fully described in the *ZeBu Server User Guide*.

The Direct ICE interface (only on ZS 1, ZS 2 and ZS 3) is available to connect the DUT to a target system or a hardware IP core via 1,200 data pins and dedicated clock pins. This interface and its usage are fully described in the *ZeBu Server User Guide* corresponding to your hardware.

1.1.1 Design Capacity

TABLE 2 Design Capacity per ZeBu Server 1 Module

ZeBu Server 1 Module	Design FPGAs	Design Capacity (ASIC gates)	DDR2 Bulk Memory	RLDRAM II Memory	ICE Pins
4C	4	6 M	2x2 GBytes	4x4x512 MBits	None
8C/ICE	8	12 M	1x2 GBytes	8x2x512 MBits	1,200
16C	16	22 M	2x2 GBytes	None	None

TABLE 3 Design Capacity per ZeBu Server 2 Module

ZeBu Server 2 Module	Design FPGAs	Design Capacity (ASIC gates)	DDR2 Bulk Memory	RLDRAM Memory	ICE Pins
5F/ICE	5	12 M	1x4 GBytes	4x 1GBits	1,200
9F/ICE	9	21 M	4x4 GBits + 1x4 GBytes	None	1,200
17F	17	40 M	1x4 GBytes	None	None

TABLE 4 Design Capacity per ZeBu Server 3 Module

ZeBu Server 3 Module	Design FPGAs	Design Capacity (ASIC gates)	DDR3 Memory	ICE Pins
9F	9	60 M	4x 512MB 2 X 8 GB	None
9F/ICE	9	60 M	4x 512MB 2 X 8 GB	568

TABLE 5 Design Capacity per ZeBu Server 4 Module

ZeBu Server 4 Module	Design FPGAs	Design Capacity (ASIC gates)	DDR3 Memory	ICE Pins
12F	12	150M	2x 1 GB on 6 FPGAs per module 1x 8 GB DDR3 on 2 FPGAs per module	None

1.2 ZeBu Server Hardware for Server 1, Server 2, and Server 3

1.2.1 2-slot ZeBu Server Unit

The ZeBu Server unit is the basic element in a ZeBu Server system. The 2-slot unit is not intended to be part of a multi unit configuration; it is always seen as the ZeBu Server system itself.

A 2-slot ZeBu Server unit consists of the following elements:

- 2 slots to plug up to 2 FPGA modules.
- A backplane equipped with the following:
 - PCIe interface: 4 connectors to link up to 4 PCs.
 - Smart Z-ICE interface: 4 connectors with 64 data pins and 4 clock pins.

1.2.1.1 Front Panel

The front panel features:

- ON/OFF switch
- SD slot accepting compact memory cards
- LCD display



FIGURE 5. Front Panel of a 2-slot Unit

1.2.1.2 Rear Panel

The rear panel features:

- **HOST** connectors (in a single-unit system):
 - C0-C3: 4 PCIe connectors to link up to 4 PCs
- **Smart Z-ICE** connectors:
 - P0-P3: 4 ERNI 50-pin connectors
- Power supply connector (standard IEC-C20 inlet)
- Fuse (10A max)



FIGURE 6. Rear Panel of a 2-slot Unit

1.2.2 5-slot ZeBu Server Unit

The ZeBu Server unit is the basic element in any ZeBu Server system. Unlike 2-slot units, the 5-slot unit can be stand-alone or part of a multi unit configuration. Therefore, it can be the ZeBu Server system itself in a single-unit configuration or part of the ZeBu Server system in a multi unit configuration. The 5-slot ZeBu Server unit consists of the following elements:

- 5 slots to plug up to 5 FPGA modules.
- A backplane equipped with the following:
 - PCIe interface (10 connectors to link up to 5 PCs, 4 units, and a hub)
 - Smart Z-ICE interface (5 connectors with 80 data pins and 5 clock pins)

1.2.2.1 Front Panels

The front panel features:

- ON/OFF switch

- SD slot accepting compact memory cards



FIGURE 7. : Front panel of a 5-slot unit

1.2.2.2 Rear Panel

The rear panel features:

- **HOST** connectors (in a single-unit system):
 - ❑ C0-C4: PCIe connectors to link up to 5 PCs
 - ❑ C5-C9: Not used
- **HOST** connectors (in a multi unit system):
- Unit U0:
 - ❑ C0-C3: PCIe connectors to link up to 4 PCs
 - ❑ C4: Connector to link a hub
 - ❑ C5-C9: to inter-connect with the other units
- Units U1-U4:
 - ❑ C0-C4: PCIe connectors to link up to 5 PCs
 - ❑ C5-C9: to interconnect with the other units
- **Smart Z-ICE** connectors:
 - ❑ P0-P4: 5 ERNI 50-pin connectors
- Power supply connector (standard IEC-C20 inlet)

- Fuse:
 - ZeBu Server 1: 12.5A max



FIGURE 8. Rear Panel of a 5-slot Unit

1.3 ZeBu Server Hardware for Server 4

For details about the ZeBu Server 4 hardware, see the *ZeBu Server 4 Site Planning Guide*.

1.4 Interconnection With Host PCs and Other Units

Note

📄 The connection of the ZeBu Server units to the host PCs and interconnections between ZeBu Server units are physically performed by authorized Synopsys personnel.

📄 DO NOT attempt to plug/unplug a cable to/from a unit. You may easily damage it.

1.4.1 For ZeBu Server 1, 2, and 3

1.4.1.1 2-slot ZeBu Server Unit

Up to 4 host PCs can be connected to a 2-slot ZeBu Server unit, via **HOST** connectors C0 through C3 located on the unit's rear panel.

However, only 2 users can use the unit at the same time.

1.4.1.2 5-slot ZeBu Server Unit

Single unit Configuration

Up to 5 host PCs can be connected to a 5-slot ZeBu Server unit, via **HOST** connectors C0 through C4 located on the unit's rear panel.

Multi unit Configuration

Up to 5 host PCs can be connected to a 5-slot ZeBu Server unit, via **HOST** connectors C0 through C4 located on the unit's rear panel. However only up to 4 PCs can be connected to Unit U0 (C4 on U0 is reserved for hub connection).

Interconnections with other units use connectors C5 through C9:

- C5-C8 to interconnect with 4 units in the same 5-unit cluster (U0-U4 or U5-U9)
- C9 (reserved for more than 5-unit ZeBu Server systems) used to interconnect with Unit N+5 or N-5 in the other 5-unit cluster (e.g. to connect U0 to U5, U9 to U4, and so on.)

1.4.2 For ZeBu Server 4

For details on ZeBu Server 4 single unit and multi unit configurations, see the *ZeBu Server 4 Site Planning Guide*.

1.5 Single Unit System Cabling

In single-unit configurations, the ZeBu Server unit is the ZeBu Server system itself.

- Up to 2 users can use a 2-slot ZeBu Server unit at the same time (even if 4 host PCs are connected to the 4 HOST connectors available on the rear panel)
- Up to 5 users can connect to a 5-slot ZeBu Server unit at the same time (via the first 5 HOST connectors of the 10 connectors available on the rear panel)



FIGURE 9. Single-unit System Cabling (2- and 5-slot units)

Note

This is applicable only for ZeBu Server 1, 2 and 3.

1.6 Multi Unit Interconnection Topology

In a multi unit system, the interconnection topology of cables between the units may impact the achievable runtime performance of the system.

Two different configurations are available:

- The standard configuration is optimized for runtime performance but requires a full modification of the cables if a unit is added in the system.
- A specific topology which supports a scalable system not requiring the modification of existing cables if a unit is added in the system.

The interconnection of units must not be modified by end-users because such modifications may damage the system. Such modifications should be done only by Synopsys personnel, as described in chapter [Hardware Installation](#).

1.7 Multi User Configurations for ZeBu Server 1, 2, and 3

The maximum number of users allowed on a ZeBu Server system depends on the ZeBu Server system configuration. It ranges from 2 to 49 users.

TABLE 6 Number of Users per ZeBu Server System

ZeBu Server System	ZeBu Server Unit IDs	HOST Connectors for PCs	Max. Users per Unit	Max. Users per System
1x 2-slot unit	U0	C0-C3	2	2
1x 5-slot unit	U0	C0-C4	5	5
2x 5-slot units	U0 + U1	C0-C4 (*)	5 (**)	9
3x 5-slot units	U0 to U2	C0-C4 (*)	5 (**)	14
4x 5-slot units	U0 to U3	C0-C4 (*)	5 (**)	19
5x 5-slot units	U0 to U4	C0-C4 (*)	5 (**)	24
6x 5-slot units	U0 to U5	C0-C4 (*)	5 (**)	29
7x 5-slot units	U0 to U6	C0-C4 (*)	5 (**)	34
8x 5-slot units	U0 to U7	C0-C4 (*)	5 (**)	39
9x 5-slot units	U0 to U8	C0-C4 (*)	5 (**)	44
10x 5-slot units	U0 to U9	C0-C4 (*)	5 (**)	49

(*) Connector C4 on U0 is reserved for the hub when multi unit system is set up.

(**) Only 4 users can connect to U0 when multi unit system is set up.

Note

You must have as many host PCs as testbenches that may run simultaneously on a ZeBu Server unit.

1.8 Multi User Configurations for ZeBu Server 4

For details on ZeBu Server 4 multi user configurations, see the *ZeBu Server 4 Site Planning Guide*.

1.9 Multi PCIe Cabling for ZeBu Server 1, 2, and 3

In a multi unit configuration, up to 5 PCIe boards can be plugged in a single host PC with each PCIe board connected to 1 or more ZeBu Server units. This multi PCIe provides an increased bandwidth for communication between the ZeBu Server units and the host PCs, when transferring a large amount of data.

TABLE 7 ZeBu Server Configuration

If your ZeBu Server configuration includes...	Then you can use:
2 units	1 or 2 PCIe boards
3 units	3 PCIe boards
4 units	4 PCIe boards
5 to 10 units	5 PCIe boards

Note

It is not possible to have the same PC connected through several PCIe boards to the same unit.

When using 3 units or more, it is mandatory to have at least 2 host PCs with their PCIe boards connected to U0. If very large designs are mapped on other units than U0, several PCIe boards may be connected in other host PCs as well.

1.10 Multi PCIe Cabling for ZeBu Server 4

For details on ZeBu Server 4 multi PCIe cabling, see the *ZeBu Server 4 Site Planning Guide*.

2 PC Requirements for Installation

The ZeBu Server hardware and software can be installed on most Linux-operated PCs. Read this chapter before installing the ZeBu Server software and hardware to ensure that the chosen PC is suitable for installation.

For details on PC requirements for ZeBu Server 4 installation, see the *ZeBu Server 4 Site Planning Guide*.

For technical data information and site installation requirements, see chapter [Appendix A: Technical Data](#).

This chapter provides information about the hardware and software requirements for ZeBu installation and consists of the following sub-sections:

- [Hardware Requirements](#)
- [Software Requirements](#)

2.1 Hardware Requirements

2.1.1 Physical Connection in the PC

The ZeBu Server interconnection board is plugged in a PCIe slot in the host PC. For debug purposes, it may be interesting to use a PC with a RESET button to restart the system without powering OFF.

You must have as many host PCs as testbenches that may run simultaneously on a ZeBu Server unit.

2.1.1.1 PC Platform Compatibility

The ZeBu compilation and runtime tools are 64-bit software which can only be used on 64-bit PC configurations.

It is recommended to use only PC configurations previously tested by Synopsys. When using another PC configuration, you may encounter malfunction during installation and at runtime. In such a case, you should see section [ZeBu Server Unit not Detected by The PC](#).

You should inform Synopsys if you want to use a PC configuration which is not part of the list of PC configurations tested and recommended by Synopsys. It is important that Synopsys has enough time to test this new PC before you use it with ZeBu Server.

The lists of recommended PC configurations for compilation and emulation runtime are available in the following sections.

Suggested Machines for Compilation

- Dell PowerEdge R620
- Dell PowerEdge R630
- HP Proliant DL360p Gen8, Gen9
- HP Proliant DL380p Gen8, Gen9

Note

Dell PowerEdge R620 cannot be used with ZS 4.

Suggested Machines for Emulation Runtime

- Dell PowerEdge R620
- HP Proliant DL360p Gen8, Gen9
- HP Z620, HP Z640
- HP Z820, Z840
- HP DL580 gen8
- HP Z420, HP Z440

Note

Dell PowerEdge R620 cannot be used with ZS 4.

For the host PC, the emulation runtime requires PC configurations with at least four cores. Additional cores are required when using advanced features in the host or when using multi-threaded verification environments.

Note

Some of the recommended models above come in various form-factors. Ensure that the form-factor can accommodate the ZeBu PCIe card.

For large multi-chassis configurations (3+), hosts that can accommodate multiple ZeBu PCIe cards are strongly recommended.

2.1.1.2 Memory Requirements

Memory requirements when using ZeBu Server vary based on the design size. For guidance, see the following table and the sections.

TABLE 8 Memory Requirements for ZeBu Server 1, 2, and 3

ZeBu Hardware	Design Size	FPGA Resources in the System	RAM Size for ZeBu Compilation	RAM Size for Runtime Database
ZeBu Server 1	12 MGates	1x 4C module	4 GBytes	1 Gbytes
	30 MGates	1x 16C module	10 GBytes	2.4 Gbytes

TABLE 8 Memory Requirements for ZeBu Server 1, 2, and 3

	200 MGates	5x 16C modules	32 GBytes	16 GBytes
	500 MGates	5 units x (5x 16C modules)	80 GBytes	40 GBytes
	1 BGates	10 units x (5x 16C modules)	160 GBytes	80 GBytes
ZeBu Server 2	12 MGates	1x 5F module	8 GBytes	1.6 Gbytes
	40 MGates	1x 17F module	20 GBytes	5.5 Gbytes
	200 MGates	5x 17F modules	32 GBytes	27.2 GBytes
	1 BGates	5 units x (5x 17F modules)	160 GBytes	136 GBytes
	2 BGates	10 units x (5x 17F modules)	256 GBytes	272 GBytes
ZeBu Server 3	60 MGates	1x 9F module	32 GBytes	16/32 Gbytes
	300 MGates	5x 9F modules	64 GBytes	32/64 GBytes
	1.5 BGates	5 units x (5x 9F modules)	128 GBytes	128 GBytes
	3 BGates	10 units x (5x 9F modules)	256 GBytes	256 GBytes

Memory Requirements for Compilation

For compilation, the size of the design impacts the memory requirements as described in [Table 8](#) and [Table 10](#), we recommend:

- 128GB RAM minimum
- 256GB RAM recommended

Memory Requirements for FPGA Place & Route

For FPGA Place & Route software, the memory requirements should be the following to avoid swapping when compiling one FPGA:

Hardware Requirements

- **ZeBu Server 1 and ZeBu Server 2:** Xilinx ISE Place & Route software requires 8 GB RAM
- **ZeBu Server 3:** Xilinx Vivado & Triton Placer for Vivado requires 16 GB RAM

Memory Requirements for Emulation Runtime

For emulation runtime, the PC must have enough memory to load the runtime data so we recommend:

- 64GB RAM minimum
- 256GB RAM for designs that require many transactors or a testbench that allocates many memories

Based on your verification environment (in particular for an HDL simulator) and the architecture of your testbench, you may need to increase the memory capacity to get the desired performance.

Farm Requirements

Regarding the compilation farm, it is recommended to have:

- In a single unit environment, at least one PC with 32 GB RAM
- In a multi unit environment, several PCs with 64 GB RAM

For any ZeBu Server configuration, it is strongly recommended to compile the design in parallel on a PC farm that has at least one PC with the memory requirements listed in [Table 8](#), instead of compiling on a single PC.

To verify the total available memory on a PC, type the following command shell: \$ free

2.1.1.3 Hard Disk Requirements

The ZeBu software, including the specific Xilinx Place & Route subset, and the additional packages (see section [Downloading the Packages for Installation](#)) require a maximum of 25 GB on your hard disk after installation.

The overall size recommended for your hard disk for ZeBu software installation is 37 GB, due to the temporary data stored during installation.

To get the total available disk space in each partition of the PC, type the following command in your shell: `$ df -k`

When compiling the design for ZeBu, the estimated necessary disk space is about 1GB per FPGA.

2.1.1.4 How to Determine the Number of Compute Servers for Compilation

The example below features a ZeBu environment comprised of Servers with 256GB RAM and 16 cores. The breakdown below is for a 15-module design.

Thread Calculation

- 15 modules x 9 FPGAs = 135 FPGAs
- 135 FPGAs x 4 threads = 540 concurrent jobs

Server Estimate

Assuming each server has 16 cores with each running 2 threads: 32 threads:

- Number of servers = $540 / 32 \approx 17$ machines
- Memory for servers = 128 GB (required)/256GB (suggested)
- Each Vivado job is expected to take 16GB independently of the number of threads
- 4 threads per Vivado job
- 32-thread machine: 8 concurrent Vivado jobs
- 112GB RAM

2.2 Software Requirements

The ZeBu software for compilation and runtime runs under the Linux operating system.

Some additional third-party tools like Xilinx ISE, Vivado and Triton Placer for Vivado must be installed to have a working configuration, as described in section [Synopsys Interoperable Technologies](#).

2.2.1 Linux Operating System Requirements

Installing and upgrading the Linux operating system is typically carried out by the system administrator.

Linux Operating System Compatibility

ZeBu requires one of the following operating systems installed in 64 bits for both compilation and emulation runtime:

- Red Hat Enterprise Linux 6.6
- SUSE Linux Enterprise Server 11, 12
- Ubuntu 14.04 LTS
- CentOS 7.3

For more information, see specific limitations mentioned in the *ZeBu Server Release Notes*.

Linux Kernel Compatibility

ZeBu requires a kernel from the 2.6 series or higher on Red Hat Enterprise Linux and SUSE Linux Enterprise Server.

However, you should check in the *ZeBu Server Release Notes* to know about which versions of the Linux kernel have been tested for the targeted ZeBu version.

To get the currently loaded kernel version of your Linux, type the following command in your shell: `$ uname -r`

It is recommended to have the source files for your kernel version installed on the PC so that you can compile the ZeBu loadable kernel module (zKernel) if required.

These source files are in an additional Linux package: `kernel-devel-<version>`.

Package Selection

The appropriate Linux packages for software development must be installed on the PC where ZeBu is used (the package names may vary based on the Linux distribution you are using), in particular:

- `freetype-devel`
- `fontconfig-devel`
- `kernel-devel-<version>` for compilation of zKernel

If there are no disk space constraints when installing the Linux operating system, it may be easier to install a complete version of it, including all the packages.

If some required packages were not installed on the PC during the initial installation of Linux operating system, they can be installed afterwards.

If you previously installed these packages in 32-bit mode, you must re-install them in 64-bit mode.

2.2.1.1 Network Access to the ZeBu Software

The absolute path to the installation directory must be identical on the PC from which the ZeBu software is installed and on the PC from which the ZeBu software will be used.

Once installed, the ZeBu software must not be copied to any other directory in the network.

2.2.1.2 License Software Compatibility

Using ZeBu requires two FLEXnet license servers to work:

- The ZeBu software uses the Synopsys Common Licensing (SCL) version 11.9 or higher.
- The Xilinx ISE, Vivado and Triton Placer for Vivado (see [Table 10](#)) uses a separate license server.

The license servers and the ZeBu software can run on the same PC or on different PCs based on your IT configuration.

For more information about the versions of ISE and Vivado supported by this ZeBu Server release, see *ZeBu Server Release Notes*.

ZeBu License Software Compatibility

For information on the software compatibility, see the [Supported SCL Platforms by Operating System & Platform Keyword](#) page from the [Synopsys Licensing QuickStart Guide](#), available on the Synopsys website.

Xilinx License Software Compatibility

The Xilinx license software runs under the following operating systems:

- 32-bit Solaris
- 32-bit Linux
- 64-bit Linux

2.2.1.3 Synopsys Interoperable Technologies

Synopsys has many technologies that work with ZeBu Server and some of its specific features. The following table shows the technologies that have been successfully tested with ZeBu Server.

For more information on the tested versions of these technologies, see the *ZeBu Server Release Notes*.

TABLE 9 Synopsys Interoperable Technologies

Tool	Description
Synplify Pro Synplify Premier Synplify Premier DP	FPGA synthesizer. Must be purchased separately. With Synplify tools, the synthesis can be performed within or outside the ZeBu flow.
VCS MX	HDL simulator. Mandatory when using the Unified Compile feature. It is mandatory to install and run VCS in 64-bit mode on 64-bit PC configurations.
MVtools	Synopsys Low Power Verification Tools Suite.
Verdi ³	Debug environment.
Siloti	Verdi ³ waveform viewer.

For more information on these technologies and the ZeBu features requiring them, see section [Downloading Required Interoperable Technologies](#).

2.2.1.4 Third-Party Tools

ZeBu also works in association with third-party tools for synthesis, compilation, co-simulation, and debug. Some of these tools are mandatory and others are left to user's choice (for example, one or other synthesis tools may be preferred).

It is recommended that third party tools be installed and run in 64-bit mode on 64-bit PC configurations; this is mandatory for runtime tools such as gcc or SystemC.

Tested Third-Party Tools

The following table gives examples of third-party tools that have been successfully tested by Synopsys.

TABLE 10 Tested Third-party Tools

Tool	Description
SystemC	Required for SystemC co-simulation. http://www.systemc.org
gcc	C compiler, part of the Linux operating system distribution. http://gcc.gnu.org/ Required for C/C++, SystemC co-simulation, transaction-based verification. May be required during installation process in case of kernel compilation.
ISE FPGA Place & Route for ZeBu Server 1 and ZeBu Server 2	Xilinx tools for FPGA Place & Route during ZeBu compilation. http://www.xilinx.com Included in ZeBu Server software package (see sections Xilinx Place & Route Software and Downloading the ZeBu Software Package).
Vivado Place & Route for ZeBu Server 3 and ZeBu Server 4	
Triton Placer for Vivado in Zebu Server 3	

Third-party tools tested with a given ZeBu software release are listed with version information in the corresponding *ZeBu Server Release Notes*.

FPGA Synthesizers

ZeBu supports the following FPGA synthesizers:

- **ZeBu Fast Synthesis (zFAST)**, which is included in ZeBu.
- **Synplify Pro, Synplify Premier and Synplify Premier DP**: Must be purchased separately from Synopsys. See [Table 9](#) for more information.
- **Other third-party synthesizers**: Must be purchased separately from their distributors. With these tools, the synthesis can only be performed out of the ZeBu flow to generate EDIF files for the ZeBu compiler. However, keep in mind they have not been tested by Synopsys.

Xilinx Place & Route Software

The ZeBu delivery package includes specific Xilinx ISE, Vivado and Triton packages. They are subsets of Xilinx software supporting FPGA Place & Route for ZeBu.

The versions of the Xilinx ISE, Vivado and Triton Placer for Vivado that have been tested for a ZeBu software version is mentioned in the corresponding *ZeBu Server Release Notes*.

Some specific patches developed by Xilinx for ZeBu may be included in this subset, thus it is not recommended to use a release obtained directly from Xilinx.

The ZeBu software has been optimized for a specific subset of Xilinx Place & Route software.

Installing the Xilinx subset for ZeBu during installation of the ZeBu software package is highly recommended.

Once installed, the ZeBu compilation settings for FPGA Place & Route force the use of this dedicated Xilinx Place & Route installation (see chapter [Configuring End-User's Environment](#)), independently of any other Xilinx installation.

3 Hardware Installation

This chapter provides information about how to install the ZeBu hardware and consists of the following sub-sections:

- [Installing the ZeBu Server PCIe Board in a Host PC](#)
- [Removing the ZeBu Server PCIe Board from a Host PC](#)
- [Connecting the ZeBu Server PCIe Cables to the Host PC](#)
- [Connecting ZeBu Server Units to the Power Supply](#)
- [Changing the Power Supply Fuse on a ZeBu Server Unit](#)
- [Getting Board Labels](#)

Hardware installation (or upgrade) of a ZeBu Server unit must be performed by Synopsys personnel. Furthermore, the ZeBu Server unit should not be moved without explicit authorization of a Synopsys representative.

It is recommended to unpack the ZeBu Server unit and leave it at room temperature for one hour before performing installation. This is particularly important if the system was shipped by air or during cold weather.

Only the following operations may be performed without the presence of Synopsys personnel:

- Installing the ZeBu Server PCIe board in a host PC, see section [Installing the ZeBu Server PCIe Board in a Host PC](#).
- Connecting the PCIe cable on the host PC, see section [Removing the ZeBu Server PCIe Board from a Host PC](#).
- Connecting the ZeBu Server unit to the Power Supply, see section [Connecting ZeBu Server Units to the Power Supply](#).


3.1 Installing the ZeBu Server PCIe Board in a Host PC


SAFETY NOTE

Installation of the PCIe board must be performed by competent personnel (with sufficient knowledge and training, and suitably equipped). Above all, the host PC and the power socket outlet must be switched OFF (wherever possible), then the power cord removed **BEFORE** you attempt to install the PCIe board inside a host PC.

Electricity can kill. Even non-fatal shocks can cause severe and permanent injury. Voltages inside the host PC are POTENTIALLY LETHAL.

Note

 *The PCIe board can become very hot.*

 *Do not attempt to plug/unplug a cable to/from a unit: you may easily damage it.*


If the host PC has some EMC protection on the PCIe slots, these should be removed before plugging the PCIe boards because they may cause issues when plugging the PCIe cables. Be careful when removing EMC protections because they are very sharp and may cause injury.


To install the ZeBu Server PCIe interconnection board in a host PC:

1. Remove all jewelry from your hands and wrists.
2. Use only insulated or non-conducting tools.
3. Switch OFF the PC and switch OFF at the power socket (if possible).
4. Unplug the PC's power cord and wait 10 to 20 seconds to allow voltage levels inside the PC to fall.
5. Remove the cover from the PC.
6. Remove the slot-cover from a vacant, unshared bus-mastering PCIe slot and save the screw.
7. Carefully insert the PCIe board in the PCIe slot.
8. Secure the board in place using the slot-cover screw removed in step 6.
9. Replace the PC cover.
10. Reconnect in the power cord to the PC.

Removing the ZeBu Server PCIe Board from a Host PC

Note

 The PCIe board is powered from the PCIe bus and the white power connector on the board **MUST NOT** be connected directly to the PC power supply.

 The connection of the ZeBu Server units to the host PCs and interconnections between ZeBu Server units are physically performed by authorized Synopsys personnel.

3.2 Removing the ZeBu Server PCIe Board from a Host PC

The PCIe board can become very hot.

If the PCIe board must be removed from a previous host PC, the following procedure should be followed:

1. Remove all jewelry from your hands and wrists.
2. Use only insulated or non-conducting tools.
3. Switch OFF the PC and switch OFF at the power socket (if possible).
4. Unplug the PC's power cord and wait 10 to 20 seconds to allow voltage levels inside the PC to fall.
5. Check and write down the cabling between the host PC and the unit. This will facilitate your next installation.
6. On the host PC, unplug the cables from the PCIe board.
7. Remove the cover from the PC.
8. Unmount the PCIe boards carefully.

3.3 Connecting the ZeBu Server PCIe Cables to the Host PC

Note

DO NOT attempt to plug/unplug a cable to/from a unit. You may easily damage it.

You may need to connect or disconnect one or several ZeBu Server PCIe cables on the PCIe board when changing a host PC in your system or adding new units to your system.

There is no need for tools to connect/disconnect the PCIe cables on the PCIe board.

For connection to the ZeBu Server unit, the PCIe board has two connectors: 1 with a blue sticker (Rx) and 1 with a red sticker (Tx). The corresponding cables are also labeled with blue and red stickers.

To connect the ZeBu Server PCIe cables to a host PC:

1. Check that the ZeBu Server units and the host PC are already switched OFF and unplugged.
2. Remove the PCIe boards from the previous host PC, as described in section [Removing the ZeBu Server PCIe Board from a Host PC](#).
3. Connect the PCIe boards in the new host PC.
4. Switch ON the host PC.
5. Launch `zInstallLauncher.sh` to get the mapping order of the PCIe boards, as described in section [PCIe Boards Detection Order](#).
6. Switch OFF the host PC; unplug the PC's power cord and wait 10 to 20 seconds to allow voltage levels inside the PC to fall.
7. Considering the correct mapping of the cables (based on the unit numbers and PCIe numbering), connect the cables to the PCIe board matching the color stickers:
 - a. Cable end with blue sticker on PCIe board connector with blue sticker;
 - b. Cable end with red sticker on PCIe board connector with red sticker.

For a good connection, the connector should be correctly locked on both left and right sides.

Connecting ZeBu Server Units to the Power Supply

If the EMC protections are present on the PCIe slot of the PC, it may be more difficult to plug the cable correctly.

The connection of the ZeBu Server units to the host PCs and interconnections between ZeBu Server units are physically performed by authorized Synopsys personnel.

3.4 Connecting ZeBu Server Units to the Power Supply

The power socket is located on the rear panel of each ZeBu Server unit in the system. Select the appropriate cord and connect it as follows:

1. Ensure that the main socket and the ZeBu Server units are all switched OFF (panel switch position on the ZeBu Server units = 0).
2. Connect the power cords to the respective ZeBu Server units.
3. Connect the power cords to the mains socket.
4. Switch ON the mains socket.
5. Switch ON the
6. ZeBu Server units (panel switch position = 1).

You can now switch ON the host PCs connected to ZeBu Server units and initialize the ZeBu Server system from the relevant PC (For more information, see chapters [ZeBu Server System Setup](#) and [Initializing the ZeBu Server System](#)).

3.5 Changing the Power Supply Fuse on a ZeBu Server Unit

When the ZeBu Server unit is shipped by Synopsys, the correct fuse is fitted. If the fuse blows or the equipment does not appear to work when switched ON, you may need to change the supply fuse. The fuse holder is located on the back panel of the ZeBu Server unit.

The requirements regarding the fuses are the following:

TABLE 11 ZeBu Server Fuse Information

ZeBu Server Version	Number of slots	Fast Acting Fuse
ZeBu Server 1	2-slot unit	20 mm 10 Amp Class T
	5-slot unit	20 mm 12.5 Amp Class T
ZeBu Server 2	2-slot unit	20 mm 10 Amp Class T
ZeBu Server 3	5-slot unit	20 mm 15 Amp Class T
ZeBu Server 4	4-slot units	20mm 15 Amp Class T

To change the fuse in the ZeBu Server unit, proceed as follows:

1. Stop, and switch OFF the entire system.
 - a. Switch OFF the ZeBu Server unit.
 - b. Switch OFF the mains socket.
2. Disconnect the ZeBu Server power cord from the mains socket.
3. Disconnect the ZeBu Server power cord from the rear panel power socket.
4. Use a small screwdriver to turn the left fuse holder barrel a quarter turn (90°) and pull the fuse holder out of the fuse block.
5. Remove the old fuse and push the new fuse into position.
6. Push the fuse holder back into position in the fuse block.
7. Reconnect the power cord as described in section [Getting Board Labels](#).

3.6 Getting Board Labels

If a Synopsys representative requests the information, for maintenance purposes, you can obtain the board labels of your ZeBu Server system:

```
zUtils -getLabel <labelBoardName> <labelFileName>
```

where:

- <labelBoardName> is any of the following boards:

PC	Hub	Unit U0	Unit U1	Unit U2	Unit U3	Unit U4
S0_PCIe	HUB	U0_BP	U1_BP	U2_BP	U3_BP	U4_BP
S1_PCIe		U0_FP	U1_FP	U2_FP	U3_FP	U4_FP
S2_PCIe		U0_M0	U1_M0	U2_M0	U3_M0	U4_M0
S3_PCIe		U0_M1	U1_M1	U2_M1	U3_M1	U4_M1
S4_PCIe		U0_M2	U1_M2	U2_M2	U3_M2	U4_M2
		U0_M3	U1_M3	U2_M3	U3_M3	U4_M3
		U0_M4	U1_M4	U2_M4	U3_M4	U4_M4

- Where BP stands for backplane and FP for front panel.
If <labelBoardName> is not specified, all labels will be dumped on screen.
- <labelFileName> is a file where the E2PROM content of the specified board will be stored. If <labelFileName> is not specified, then the board label will be dumped on screen.

Note

To get the label of a PCIe board plugged in a given host PC, you must be connected to the same host PC. You cannot get the labels of other PCIe boards or even know what other PCs are connected to the system.

The output depends on whether the host PC from which the command was issued is connected to unit U0:

- If the host PC is connected to U0:
- Output = Full label (for any board). Example for ZeBu Server 1:

```
-- ZeBu : zUtils : ==== Unit 0 Module 0 label ====  
$labelVersion = 2;  
$boardType    = "zse_xc6v_4c_ice_lx760_v3";  
$initDay      = "2-2-2015";  
$id           = 0x0102;  
$revPCB       = "3.0";  
$revBOM       = "2.0";  
$features     = 0xffffffff;
```

- If the host PC is not connected to U0:
 - ❑ Output = Full label (for the PCIe board plugged in the current host PC).
 - ❑ Output = Short label (for any other board). Example for ZeBu Server 1:

```
-- ZeBu : zUtils : ==== Unit 4 Module 0 (physical Unit 0 Module 0)  
label ====  
$boardType    = "zse_xc6v_4c_ice_lx760_v3";  
$id           = 0x0102;
```

4 Software Installation

This chapter provides information about how to install the ZeBu software and consists of the following sub-sections:

- [*Read This First*](#)
- [*Downloading the Packages for Installation*](#)
- [*Installing the ZeBu Software*](#)
- [*Diagnostics Patches Overview*](#)
- [*Post- Installation*](#)
- [*Installing Licenses for ZeBu*](#)
- [*Downloading Required Interoperable Technologies*](#)

You can install the ZeBu software on the PC to which the ZeBu Server unit is connected or on a PC which is not physically connected to the ZeBu Server unit. The absolute path to the installation directory must be identical on the PC from which the ZeBu software is installed and on the PC from which the ZeBu software is used.

Once installed, the ZeBu software must not be copied to any other directory in the network. If you want to re-install the ZeBu software, then installing in a new directory is preferable.

4.1 Read This First

Before proceeding with software installation, read this summary, and make sure you understand the various steps from the installation of the ZeBu software to the initialization of the ZeBu Server system:

1. Install the ZeBu software (this chapter).
2. Configure the end-user's environment (see chapter [Configuring End-User's Environment](#)).
3. Run `zInstallLauncher.sh` on each host PC connected to the ZeBu Server system. This will initialize the PCIe board and load `zKernel` in the PC's operating system (see chapter [Initializing the Host PCs](#)).
4. Perform a system setup for the entire ZeBu Server system (see chapter [ZeBu Server System Setup](#)).
5. Initialize the ZeBu Server system:

❑ After each configuration change:

Run `zSetupSystem <my_setup>.zini` from any PC connected to U0. This will launch system calibration and diagnostics, test the memories, and generate a configuration file (see chapter [ZeBu Server System Setup](#)).

❑ After each power OFF/ON of any unit in the system:

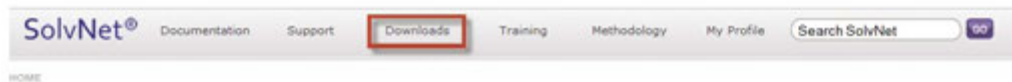
Run `zUtils -initSystem` from any PC connected to unit U0. This will load the backplanes and the modules' system FPGAs (see section [Initializing the System With zUtils -initSystem](#)).

4.2 Downloading the Packages for Installation

4.2.1 Downloading and extracting the Synopsys Installer

Before installing the ZeBu software, it is mandatory to first download the **Synopsys Installer** from SolvNet and extract it.

1. Log on the Synopsys SolvNet website (<https://solvnet.synopsys.com>) with your credentials.
2. Click the **Downloads** link at the top of the page.



3. In the **Downloads** list, choose **Synopsys Installer** then choose the latest version available.
4. In the next page, click **Download Here**.
The **Electronic Software Transfer** terms and conditions are displayed.
5. Click **YES, I AGREE TO THE ABOVE TERMS** at the end of **Electronic Software Transfer** terms. The **Synopsys Electronic Software Transfer (EST)** page opens.
6. Download the self-extracting file **SynopsysInstaller_v4.1.run** containing the **Synopsys Installer** software to a dedicated directory, for instance `/home/<user>/install_files`.

NOTE: For more details, see *Synopsys Installer on SolvNet*.

7. Once the file is downloaded, launch the following command to extract its content:

```
$ cd /home/<user>/install_files
$ SynopsysInstaller_v4.1.run
```

The **Synopsys Installer** script prompts you to specify the directory where it should be extracted.

Note

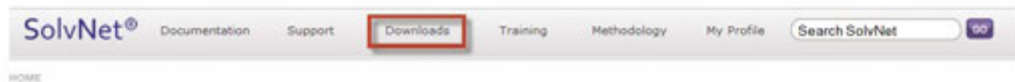
You can also add the `-d` option to the command above to specify the path where the Synopsys Installer should be extracted.

For more information, consult the downloadable file `zebu_installer_INSTALL_README.txt` available on the **Synopsys Electronic Software Transfer (EST)** page.

4.2.2 Downloading the ZeBu Software Package

Once the **Synopsys Installer** software is downloaded and extracted, you must download the ZeBu Software Package from SolvNet.

1. Log on the Synopsys SolvNet website (<https://solvnet.synopsys.com>) with your credentials.
2. Click the **Downloads** link at the top of the page.



3. In the **Downloads** list, choose **ZeBu Server (for all hardware) > 0-2018.09**.
4. In the next page, click **Download Here**.

The **Electronic Software Transfer terms and conditions** are displayed.

5. Click **YES, I AGREE TO THE ABOVE TERMS** at the end of **Electronic Software Transfer** terms.

The **Synopsys Electronic Software Transfer (EST)** page opens.

6. Download all the files required for the installation of the ZeBu software release:

- ☐ `zebu_INSTALL_README.txt`
- ☐ `zebu_vM-2017.03_linux64.spf`
- ☐ `zebu_vM-2017.03_common.spf`

7. Download the **ZeBu Device Driver Switching Solution** package that contains the following drivers.

- ☐ For ZeBu Server 1/ZeBu Server 2/ZeBu Server 3, visit, <https://solvnet.synopsys.com/retrieve/2648882.html>:

◆

 Downloading the Packages for Installation

- ❑ For ZeBu Server 4, visit <https://solvnet.synopsys.com/retrieve/2648858.html>:



For more information, see chapter [Initializing the Host PCs](#) and chapter [Appendix B: ZeBu Device Driver Switching Solution](#).

8. Download the following files containing the Xilinx Place & Route software based on your hardware:

- ❑ **Xilinx ISE Place & Route for ZeBu Server 1 and ZeBu Server 2:**
zebu_xilinx_ise_13_4_patched_vO-2018.09_common.spf

- ❑ **Xilinx Vivado Place & Route for ZeBu Server 3 and ZeBu Server 4:**
zebu_xilinx_vivado_2018_1_vO-2018.09_common.spf

- ❑ **Xilinx Triton Placer & Route for ZeBu Server 3:**
zebu_xilinx_triton_2017_1_sp42F_vO-2018.09_common.spf

9. Download the ZeBu hardware diagnostics package(s) matching your ZeBu configuration: zebu_<hardware_version>_<sys_config>_vM-2018.09_common.spf

For more information on the diagnostics packages, see section [Diagnostics Patches Overview](#).

Once you have downloaded all the files, move them to the directory containing the Synopsys Installer software you have previously downloaded in section [Downloading and extracting the Synopsys Installer](#) (for example, in the /home/<user>/install_files directory).

Note

To reduce disk space usage, you should download only packages matching your requirements. To view the packages that, you must download for your ZeBu system, log onto your SolvNet account.

4.3 Installing the ZeBu Software

Once you have downloaded all the required files in the `/home/<user>/install_files` directory, you can launch the installation of the ZeBu software.

The installation process can be used in a similar way when you install the ZeBu software for the first time and for modification of an already installed release (for example, to install a diagnostics patch).

Installing the ZeBu software and the Xilinx Place & Route software can be done with user privileges but appropriate write accesses must be granted on the disk space where the software will be installed.

Note

Before launching the installation, make sure the `DISPLAY` variable is correctly set in your installation environment.

4.3.1 In Graphical Mode

1. Open a terminal and launch the following commands:

```
$ cd /home/<user>/install_files  
$ ./installer -gui -new_installer
```

The graphical interface of the **Synopsys Installer** is displayed.

2. Click **Start**.
3. Enter your site information, and click **Next**.
4. Enter or browse the path to the directory where you have downloaded the ZeBu software release, the Xilinx Place & Route software and the required diagnostics package(s), then click **Next**.

The **Synopsys Installer** extracts the content of the ZeBu software release in a temporary directory `snps_installer_temp_<username>`.

5. Enter or browse the path to the installation directory and click **Next**.
6. In the product and release selection window, perform the following steps:
 - a. Select the **ZeBu Server 1 /ZeBu Server 2 /ZeBu Server 3/ZeBu Server 4/ZeBu Blade 2 Emulation** product (marked as **STAND-ALONE**).

Installing the ZeBu Software

- b. Select the Xilinx Place & Route software based on your hardware (marked as **OVERLAY**):
 - ♦ For ZeBu Server 1 and ZeBu Server 2: **Xilinx ise 13.4_patched for ZeBu**
 - ♦ For ZeBu Server 3 and ZeBu Server 4: **Xilinx Vivado 2018.1 for ZeBu®**
 - ♦ For ZeBu Server 3: **Xilinx Triton 2017.1-sp42F for ZeBu®**
 - ♦
7. Select the diagnostics package corresponding to your hardware (marked as **OVERLAY**).
8. Click **Next**. Several pop-up windows with the Xilinx license agreement are displayed.
In each window, click **I have read the license agreement**, and click **I Accept**.
9. For all the products that you want to install, confirm, or modify in the next screen:
 - ☐ The installation path
 - ☐ The platform and operating system matching your environment
10. Click **Next**.
11. If you install the Xilinx Place & Route software along with the ZeBu software, a pop-up window states that Place & Route software (**OVERLAY** product) is installed in the same directory as the ZeBu Server software (**STAND-ALONE** product). Click **Yes** to continue.
NOTE: *If you click **No**, a warning pop-up window appears stating that you cannot install the item. The installation is then stopped.*
12. If you install one or several diagnostics packages along with the ZeBu Server software, a pop-up window states that the diagnostics package(s) (**OVERLAY** product) is installed in the same directory as the ZeBu Server software (**STAND-ALONE** product). Click **Yes** to continue.
NOTE: *If you click **No**, a warning pop-up window appears stating that you cannot install the item. The installation is then stopped.*
13. In the summary screen, review the installation parameters and click **Accept**, **Install** to proceed to the installation.
14. Click **Finish** to exit the Synopsys Installer.
A **Release Note** window appears with information on:
 - ☐ The Synopsys Common Licensing

- ❑ Post-installation requirements

15. Click **Dismiss** to exit.

The installation is completed.

4.3.2 In Script Mode

- Open a terminal and launch the following commands:

```
$ cd /home/<user>/install_files  
$ ./installer
```

The script mode of the **Synopsys Installer** is started. You must fill in all the requested parameters for the installation.

Note

Unlike the graphical mode described in Section [In Graphical Mode](#), the script mode does not allow you to install the ZeBu software (STAND-ALONE product) along with the Xilinx Place & Route software and the diagnostics packages simultaneously (OVERLAY products). You must end the installation AFTER the ZeBu software.

4.3.2.1 Example of Configuration Script For The ZeBu Server Software Installation

```
Synopsys(R) Installer  
Version 4.1  
Copyright (C) 2018 Synopsys, Inc.  
All Rights Reserved  
Instructions: The list within {} shows the choices for a given  
option.
```

Installing the ZeBu Software

The entry within [] shows the default selection when you press the Return key. You can cancel the installation by entering quit when prompted for input.

Your site ID number is in the upper-right corner of your Synopsys license

key certificate. If you have trouble locating it, contact your Synopsys

representative.

Site ID number [XXXXXX]:

The site administrator is your site's main contact for Synopsys licensing and

other tool issues. You can leave your own name or type a different name.

Site administrator [John Doe]:

The contact information is the phone number and/or e-mail address of the site administrator.

Site contact information [jdoe@synopsys.com]:

Enter the path to the source directory containing the EST (ftp) or CD/DVD-ROM product files for this version [/home/<user>/install_files]:

Installer: Extracting release information from zebu_xilinx_vivado_2018_1_v0-2018.09_common.spf ...

Installer: Extracting release information from zebu_xilinx_vivado_2018_1_v0-2018.09_common.spf successfully

Installer: Extracting release information from zebu_zsl_93_161_v0-2018.09_common.spf ...

```
Installer: Extracting release information from zebu_zs1_93_161_v0-2018.09_common.spf successfully
Installer: Extracting release information from zebu_zs3_2s_v0-2018.08_common.spf ...
Installer: Extracting release information from zebu_zs3_2s_v0-2018.09_common.spf successfully
Installer: Extracting release information from zebu_zs2_2_5u_v0-2018.09_common.spf ...
Installer: Extracting release information from zebu_zs2_2_5u_v0-2018.09_common.spf successfully
Installer: Extracting release information from zebu_zs2_2s_v0-2018.09_common.spf ...
Installer: Extracting release information from zebu_zs2_2s_v0-2018.09_common.spf successfully
Installer: Extracting release information from zebu_zs1_2_3u_v0-2018.09_common.spf ...
Installer: Extracting release information from zebu_zs1_2_3u_v0-2018.09_common.spf successfully
Installer: Extracting release information from zebu_zs1_2s_v0-2018.09_common.spf ...
Installer: Extracting release information from zebu_zs1_2s_v0-2018.09_common.spf successfully
Installer: Extracting release information from zebu_zs3_2_10u_v0-2018.09_common.spf ...
Installer: Extracting release information from zebu_zs3_2_10u_v0-2018.09_common.spf successfully
Installer: Extracting release information from zebu_v0-2018.09_common.spf ...
Installer: Extracting release information from zebu_v0-2018.09_common.spf successfully
Installer: Extracting release information from zebu_zs2_6_10u_v0-2018.09_common.spf ...
Installer: Extracting release information from zebu_zs2_6_10u_v0-2018.09_common.spf successfully
```

Installing the ZeBu Software

```
Installer: Extracting release information from zebu_zs2_6_10u_vO-2018.09_common.spf successfully
Installer: Extracting release information from zebu_zb2_16c_vO-2018.09_common.spf ...
Installer: Extracting release information from zebu_zb2_16c_vO-2018.09_common.spf successfully
Installer: Extracting release information from zebu_zs2_5s_vO-2018.09_common.spf ...
Installer: Extracting release information from zebu_zs2_5s_vO-2018.09_common.spf successfully
Installer: Extracting release information from zebu_zs3_5s_vO-2018.09_common.spf ...
Installer: Extracting release information from zebu_zs3_5s_vO-2018.09_common.spf successfully
Installer: Extracting release information from zebu_xilinx_triton_2017_1_sp42F_vO-2018.09_common.spf ...
Installer: Extracting release information from zebu_xilinx_triton_2017_1_sp42F_vO-2018.09_common.spf successfully
Installer: Extracting release information from zebu_vO-2018.09_linux64.spf ...
Installer: Extracting release information from zebu_vO-2018.09_linux64.spf successfully
Installer: Extracting release information from zebu_zs1_43_vO-2018.09_common.spf ...
Installer: Extracting release information from zebu_zs1_43_vO-2018.09_common.spf successfully
Installer: Extracting release information from zebu_zb2_vO-2018.09_common.spf ...
Installer: Extracting release information from zebu_zb2_vO-2018.09_common.spf successfully
Installer: Extracting release information from zebu_zs1_4_10u_vO-2018.09_common.spf ...
Installer: Extracting release information from zebu_zs1_4_10u_vO-2018.09_common.spf successfully
```

```
Installer: Extracting release information from zebu_zs1_91_vO-2018.09_common.spf ...
Installer: Extracting release information from zebu_zs1_91_vO-2018.09_common.spf successfully
Installer: Extracting release information from
zebu_xilinx_ise_13_4_patched_vO-2018.09_common.spf ...
Installer: Extracting release information from
zebu_xilinx_ise_13_4_patched_vO-2018.09_common.spf successfully
```

The following version are available under /home/<user>/install_files:

```
[ 1] O-2018.09-1 - (zebu_zb2_16c zebu_zs1_91
zebu_xilinx_vivado_2018_1 zebu_zs1_93_161
zebu_xilinx_ise_13_4_patched zebu_zs1_2s zebu zebu_zs1_43
zebu_zs3_2_10u zebu_zs3_5s zebu_zs2_2_5u zebu_zs3_2s
zebu_xilinx_triton_2017_1_sp42F zebu_zs1_4_10u zebu_zs2_6_10u
zebu_zs2_5s zebu_zs2_2s zebu_zs1_2_3u zebu_zb2)
```

Select a version to install [1]: 1

Release selected: O-2018.09-1

Select Synopsys product(s) to install:

```
[ 1] zebu - ZeBu (R) Server 1 / ZeBu (R) Server 2 / Zebu
(R) Server 3 / ZeBu (R) Blade 2 Emulation (STAND-ALONE)
```

```
[ 2] zebu_xilinx_ise_13_4_patched - Xilinx ise
13.4_patched for ZeBu (R) (OVERLAY)
```

```
[ 3] zebu_xilinx_triton_2017_1_sp42F - Xilinx triton-
2017.1-sp42F for ZeBu (R) (OVERLAY)
```

```
[ 4] zebu_xilinx_vivado_2018_1 - Xilinx Vivado 2018.1 for
ZeBu (R) (OVERLAY)
```


Installing the ZeBu Software

```
[ 5] zebu_zb2 - ZeBu (R) Blade 2 diagnostics package
(OVERLAY)

[ 6] zebu_zb2_16c - ZeBu (R) Blade 2 diagnostics package
(OVERLAY)

[ 7] zebu_zs1_2_3u - ZeBu (R) Server 1 Multi-Unit
diagnostics package (OVERLAY)

[ 8] zebu_zs1_2s - ZeBu (R) Server 1 2 Slots diagnostics
package (OVERLAY)

[ 9] zebu_zs1_43 - ZeBu (R) Server 1 5 Slots diagnostics
package (OVERLAY)

[10] zebu_zs1_4_10u - ZeBu (R) Server 1 Multi-Unit
diagnostics package (OVERLAY)

[11] zebu_zs1_91 - ZeBu (R) Server 1 5 Slots diagnostics
package (OVERLAY)

[12] zebu_zs1_93_161 - ZeBu (R) Server 1 5 Slots
diagnostics package (OVERLAY)

[13] zebu_zs2_2_5u - ZeBu (R) Server 2 Multi-Unit
diagnostics package (OVERLAY)

[14] zebu_zs2_2s - ZeBu (R) Server 2 2 Slots diagnostics
package (OVERLAY)

[15] zebu_zs2_5s - ZeBu (R) Server 2 5 Slots diagnostics
package (OVERLAY)

[16] zebu_zs2_6_10u - ZeBu (R) Server 2 Multi-Unit
diagnostics package (OVERLAY)

[17] zebu_zs3_2_10u - ZeBu (R) Server 3 Multi-Unit
diagnostics package (OVERLAY)

[18] zebu_zs3_2s - ZeBu (R) Server 3 2 Slots diagnostics
package (OVERLAY)

[19] zebu_zs3_5s - ZeBu (R) Server 3 5 Slots diagnostics
package (OVERLAY)

[ b] back - Back to Select Another
Version
```

```
Enter list of products to install (or just enter "b" to go back to
select another version)[1]: 1
```

```
Platform(s) selected: linux64
```

```
Enter the full path to the directory where you want to install
Synopsys O-2018.09 products.  If the directory does not exist,
it will be created. [/usr/synopsys/O-2018.09]: /home/<user>/
install_files/installation_directory
```

Here is your final selection for installing Synopsys Tools:

VERSION: O-2018.09

PRODUCTS: zebu

PLATFORMS: linux64

Synopsys Media Directory (from): /home/<user>/install_files

Synopsys Install Directory (to): /home/<user>/install_files/
installation_directory

Platform-Independent Package(s) for : zebu

Disk space required: 8768 MB

Disk space available: 115640 MB

Licensed Products communicate with Synopsys servers for the purpose
of providing

software updates, detecting software piracy and verifying that
customers are using

Licensed Products in conformity with the applicable License Key for
such Licensed

Installing the ZeBu Software

```
Products. Synopsys will use information gathered in connection with  
this process
```

```
to deliver software updates and pursue software pirates and  
infringers.
```

```
-----  
-----
```

```
If the information is correct, continue with the installation.
```

```
Accept, Install? [yes]: yes
```

```
Wait while cksum is being verified.
```

```
Installer: Extracting zebu_vO-2018.09_common.spf ...
```

```
Installer: Extracting zebu_vO-2018.09_linux64.spf ...
```

```
Installing platform_independent code for product zebu
```

```
Installing linux64 code for product zebu
```

```
Run postinstallation scripts
```

```
Generate zebu environment files...          done
```

```
Create site_info...
```

```
Installation has finished successfully.
```

```
Entry "b" to go back to install another product or any other key/  
Return to quit[: q
```

```
Synopsys tools require that a supported version of Synopsys Common  
Licensing (SCL) be installed and serving the necessary licenses.  
For information on how to obtain SCL, or your license key file,  
see http://www.synopsys.com/keys
```

```
For any postinstallation setup requirements, see the product-  
specific  
chapters in the Installation Guide at http://www.synopsys.com/  
install
```

4.3.2.2 Example of Configuration Script For Xilinx Place & Route Software Installation

To install the Xilinx Place & Route Software, perform the following steps.

1. At the end of the ZeBu software installation, press **b** at the step in the following script:

```
Installation has finished successfully.  
Entry "b" to go back to install another product or any other key/  
Return to quit[]:
```

2. Relaunch the Synopsys Installer in script mode:

```
Select Synopsys product(s) to install:  
  
[ 1] zebu - ZeBu (R) Server 1 / ZeBu (R) Server 2 / Zebu  
(R) Server 3 / ZeBu (R) Blade 2 Emulation (STAND-ALONE)  
[ 2] zebu_xilinx_ise_13_4_patched - Xilinx ise  
13.4_patched for ZeBu (R) (OVERLAY)  
[ 3] zebu_xilinx_triton_2017_1_sp42F - Xilinx triton-  
2017.1-sp42F for ZeBu (R) (OVERLAY)  
[ 4] zebu_xilinx_vivado_2018_1 - Xilinx Vivado 2018.1 for  
ZeBu (R) (OVERLAY)
```

Installing the ZeBu Software

```
[ 5] zebu_zb2 - ZeBu (R) Blade 2 diagnostics package
(OVERLAY)

[ 6] zebu_zb2_16c - ZeBu (R) Blade 2 diagnostics package
(OVERLAY)

[ 7] zebu_zs1_2_3u - ZeBu (R) Server 1 Multi-Unit
diagnostics package (OVERLAY)

[ 8] zebu_zs1_2s - ZeBu (R) Server 1 2 Slots diagnostics
package (OVERLAY)

[ 9] zebu_zs1_43 - ZeBu (R) Server 1 5 Slots diagnostics
package (OVERLAY)

[10] zebu_zs1_4_10u - ZeBu (R) Server 1 Multi-Unit
diagnostics package (OVERLAY)

[11] zebu_zs1_91 - ZeBu (R) Server 1 5 Slots diagnostics
package (OVERLAY)

[12] zebu_zs1_93_161 - ZeBu (R) Server 1 5 Slots
diagnostics package (OVERLAY)

[13] zebu_zs2_2_5u - ZeBu (R) Server 2 Multi-Unit
diagnostics package (OVERLAY)

[14] zebu_zs2_2s - ZeBu (R) Server 2 2 Slots diagnostics
package (OVERLAY)

[15] zebu_zs2_5s - ZeBu (R) Server 2 5 Slots diagnostics
package (OVERLAY)

[16] zebu_zs2_6_10u - ZeBu (R) Server 2 Multi-Unit
diagnostics package (OVERLAY)

[17] zebu_zs3_2_10u - ZeBu (R) Server 3 Multi-Unit
diagnostics package (OVERLAY)

[18] zebu_zs3_2s - ZeBu (R) Server 3 2 Slots diagnostics
package (OVERLAY)

[19] zebu_zs3_5s - ZeBu (R) Server 3 5 Slots diagnostics
package (OVERLAY)

[ b] back - Back to Select Another Version
```

```
Enter list of products to install (or just enter "b" to go back to
select another version)[1]: 4
Accepting zebu_xilinx_vivado_2018_1 End User License Agreement is
required, [[D]isplay|[Q]uit]?
...
Do you accept the zebu_xilinx_vivado_2018_1 End User License
Agreement?[[A]ccept|[D]ecline]:A

Accepting zebu_xilinx_vivado_2018_1 End User License Agreement is
required, [[D]isplay|[Q]uit]?
...
Do you accept the zebu_xilinx_vivado_2018_1 End User License
Agreement?[[A]ccept|[D]ecline]:A

Product(s) selected: zebu_xilinx_vivado_2018_1

Enter the full path to the directory where you want to install
Synopsys O-2018.09 products. If the directory does not exist,
it will be created. [/home/<user>/install_files/
installation_directory]:

Here is your final selection for installing Synopsys Tools:

VERSION:      O-2018.09-1
PRODUCTS:     zebu_xilinx_vivado_2018_1
PLATFORMS:
Synopsys Media Directory (from): /home/<user>/install_files
Synopsys Install Directory (to): /home/<user>/install_files/
installation_directory
```

Installing the ZeBu Software

```
Platform-Independent Package(s) for :
zebu_xilinx_vivado_2018_1

Disk space required:          10699 MB
Disk space available:         104323 MB

-----

Licensed Products communicate with Synopsys servers for the purpose
of providing
software updates, detecting software piracy and verifying that
customers are using
Licensed Products in conformity with the applicable License Key for
such Licensed
Products. Synopsys will use information gathered in connection with
this process
to deliver software updates and pursue software pirates and
infringers.

-----

If the information is correct, continue with the installation.
Accept, Install? [yes]: yes

Wait while cksum is being verified.

Installer: Extracting zebu_xilinx_vivado_2018_1_v0-
2018.09_common.spf ...

Installing platform_independent code for product
zebu_xilinx_vivado_2018_1

Installation has finished successfully.
```

```
Entry "b" to go back to install another product or any other key/  
Return to quit[]:
```

```
Synopsys tools require that a supported version of Synopsys Common  
Licensing (SCL) be installed and serving the necessary licenses.  
For information on how to obtain SCL, or your license key file,  
see http://www.synopsys.com/keys
```

```
For any postinstallation setup requirements, see the product-  
specific chapters in the Installation Guide at http://  
www.synopsys.com/install
```

4.3.2.3 Example of Configuration Script for a Diagnostics Package Installation

To install a diagnostics package, perform the following steps.

1. At the end of the ZeBu software installation, press b at the step in the following script:

```
Installation has finished successfully.  
Entry "b" to go back to install another product or any other key/  
Return to quit[]:
```

2. Relaunch the Synopsys Installer in script mode:

```
Select Synopsys product(s) to install:  
    [ 1] zebu - ZeBu (R) Server 1 / ZeBu (R) Server 2 / Zebu  
(R) Server 3 / ZeBu (R) Blade 2 Emulation (STAND-ALONE)  
    [ 2] zebu_xilinx_ise_13_4_patched - Xilinx ise  
13.4_patched for ZeBu (R) (OVERLAY)
```


Installing the ZeBu Software

```
[ 3] zebu_xilinx_triton_2017_1_sp42F - Xilinx triton-
2017.1-sp42F for ZeBu (R) (OVERLAY)

[ 4] zebu_xilinx_vivado_2018_1 - Xilinx Vivado 2018.1 for
ZeBu (R) (OVERLAY)

[ 5] zebu_zb2 - ZeBu (R) Blade 2 diagnostics package
(OVERLAY)

[ 6] zebu_zb2_16c - ZeBu (R) Blade 2 diagnostics package
(OVERLAY)

[ 7] zebu_zs1_2_3u - ZeBu (R) Server 1 Multi-Unit
diagnostics package (OVERLAY)

[ 8] zebu_zs1_2s - ZeBu (R) Server 1 2 Slots diagnostics
package (OVERLAY)

[ 9] zebu_zs1_43 - ZeBu (R) Server 1 5 Slots diagnostics
package (OVERLAY)

[10] zebu_zs1_4_10u - ZeBu (R) Server 1 Multi-Unit
diagnostics package (OVERLAY)

[11] zebu_zs1_91 - ZeBu (R) Server 1 5 Slots diagnostics
package (OVERLAY)

[12] zebu_zs1_93_161 - ZeBu (R) Server 1 5 Slots
diagnostics package (OVERLAY)

[13] zebu_zs2_2_5u - ZeBu (R) Server 2 Multi-Unit
diagnostics package (OVERLAY)

[14] zebu_zs2_2s - ZeBu (R) Server 2 2 Slots diagnostics
package (OVERLAY)

[15] zebu_zs2_5s - ZeBu (R) Server 2 5 Slots diagnostics
package (OVERLAY)

[16] zebu_zs2_6_10u - ZeBu (R) Server 2 Multi-Unit
diagnostics package (OVERLAY)

[17] zebu_zs3_2_10u - ZeBu (R) Server 3 Multi-Unit
diagnostics package (OVERLAY)

[18] zebu_zs3_2s - ZeBu (R) Server 3 2 Slots diagnostics
package (OVERLAY)

[19] zebu_zs3_5s - ZeBu (R) Server 3 5 Slots diagnostics
package (OVERLAY)
```

```
[ b] back - Back to Select Another Version
Enter list of products to install (or just enter "b" to go back to
select another version)[4]: 18

Product(s) selected: zebu_zs3_2s

Enter the full path to the directory where you want to install
Synopsys O-2018.09 products.  If the directory does not exist,
it will be created. [/home/<user>/install_files/
installation_directory]:

-----

Here is your final selection for installing Synopsys Tools:

VERSION:      O-2018.09
PRODUCTS:     zebu_zs3_2s
PLATFORMS:

Synopsys Media Directory (from): /home/<user>/install_files
Synopsys Install Directory (to): /home/<user>/install_files/
installation_directory

Platform-Independent Package(s) for : zebu_zs3_2s

Disk space required:      165 MB
Disk space available:     89179 MB
```

Installing the ZeBu Software

```
-----  
-----  
Licensed Products communicate with Synopsys servers for the purpose  
of providing  
software updates, detecting software piracy and verifying that  
customers are using  
Licensed Products in conformity with the applicable License Key for  
such Licensed  
Products. Synopsys will use information gathered in connection with  
this process  
to deliver software updates and pursue software pirates and  
infringers.  
-----  
-----  
If the information is correct, continue with the installation.  
Accept, Install? [yes]: yes  
  
Wait while cksum is being verified.  
  
Installer: Extracting zebu_zs3_2s_v0-2018.09_common.spf ...  
  Installing platform_independent code for product zebu_zs3_2s  
Installation has finished successfully.  
  
Entry "b" to go back to install another product or any other key/  
Return to quit[]:  
  
Synopsys tools require that a supported version of Synopsys Common  
Licensing (SCL) be installed and serving the necessary licenses.  
For information on how to obtain SCL, or your license key file,  
see http://www.synopsys.com/keys
```

For any postinstallation setup requirements, see the product-specific chapters in the Installation Guide at <http://www.synopsys.com/install>

4.4 Diagnostics Patches Overview

Diagnostics patches are specific to the architecture of the ZeBu system (number and types of FPGA modules in the system) and to each release. The diagnostics tool uses them, `zutils`, to check your system once installed (typical use is described in chapter [ZeBu Server System Setup](#)).

Diagnostics patches are grouped into diagnostics packages, described in section [Diagnostics Packages](#). However, for ZeBu Server 4, the diagnostics patches are provided in the release package.

For the targeted ZeBu version, you should check the *ZeBu Server Release Notes* for a full list of available diagnostics packages and patches.

4.4.1 Diagnostics Packages


Diagnostics patches are grouped into diagnostics packages with a `.spf` extension such as:


`zebu_<hardware_version>_<sys_config>_v0-2018.09_common.spf`

where:

- `<hardware_version>` is the ZeBu hardware version
- `<sys_config>` is the description of the ZeBu system
- `<vxxxx.xx>` is the ZeBu software version

Note

 To reduce disk space usage, you should download only the diagnostics package(s) matching your ZeBu configuration.

 For ZeBu Server 4, there is no diagnostics package as the diagnostics patches are provided in the release package.

4.4.1.1 Example: 5-slot 1-unit ZeBu Server 1 Systems With 8C/ICE Version 1

```
zebu_zs1_91_vO-2018.09_common.spf
```

where:

- zs1 is ZeBu Server 1
- 91→ 9: 9F/ICE module, 1: module board version
- vO-2018.09 is the ZeBu Server O-2018.09 version

4.4.1.2 Example: 5-slot 1-unit ZeBu Server 1 Systems With 8C/ICE Version 3 and 16C

```
zebu_zs1_93_161_vO-2018.09_common.spf
```

where:

- zs1 is ZeBu Server 1
- 93→ 9: 9F/ICE module, 3: module board version
- 161 → 16: 17F module, 1: module board version
- vO-2018.09 is the ZeBu Server O-2018.09 version

4.4.1.3 Example: 2-slot Unit ZeBu Server 2 in Single-unit Configuration

```
zebu_zs2_2s_vO-2018.09_common.spf
```

where:

- zs2 is ZeBu Server 2
- 2s is 2-slot unit
- vO-2018.09 is the ZeBu Server O-2018.09 version

4.4.1.4 Example: 5-slot Unit ZeBu Server 3 in Single and Multi Unit Configurations

zebu_zs3_5s_vO-2018.09_common.spf

where:

- zs3 is ZeBu Server 3
- 5s is 5-slot unit
- vO-2018.09 is the ZeBu Server O-2018.09 version

For the targeted ZeBu version, you should check the ZeBu Server Release Notes for a full list of available diagnostics packages and patches.

4.4.2 Diagnostics Patches Description for ZeBu Server 1

Diagnostics patches for ZeBu Server 1 are typically installed in
<installation_directory>/etc/firmwares/ZSE/dt/V5.0

4.4.2.1 Example of 2-slot Single-unit ZeBu Server 1 System

202_43_43.diag

It corresponds to your system configuration:

- 202 → 2: 2 slots, 0: single-unit, 2: unit version
- 43 → 4: 4C module, 3: board version

4.4.2.2 Example of 5-slot Single-unit ZeBu Server 1 System With 8C/ICE Modules:

501_91_91_91_91_91.diag

It corresponds to your system configuration:

- 501 → 5: 5 slots, 0: single-unit, 1: unit version
- 91 → 9: 8C/ICE module, 1: board version

4.4.2.3 Example of 5-slot Single Unit ZeBu Server 1 System With Loopback Modules

501_161_161_161_02_02.diag

It corresponds to your system configuration:

- 501 → 5: 5 slots, 0: single-unit, 1: unit version
- 161 → 16: 16C module, 1: board version
- 02 → 0: loopback module, 2: board version

4.4.2.4 Example of Multi Unit ZeBu Server 1 System (with up to 5 units)

C00-02-512_161_161_161_161_161-512_161_161_161_161.diag

It corresponds to your system configuration:

- C00-02 → <Cxx-yz>, where:
 - Cxx → Cable configuration label (C00 to C99)
 - y → Id of the unit where the hub is plugged (0: unit Id; default unit: U0)
 - z → Hub version (2: hub version)

NOTE: C00-00: *multi unit system in a 1-unit configuration (-00 → no hub).*

- 512 → 5: 5 slots, 1: multi unit, 2: unit version
- 161 → 16: 16C module, 1: board version

4.4.2.5 Example of Multi Unit ZeBu Server 1 System (with more than 5 units)

C02-10U_84481.diag

It corresponds to your system configuration:

- C02 → Cable configuration label (C00 to C99)
- 10U → Number of units
- 84481 → Interconnection topology

NOTE: *In the last example (configurations with more than 5 units), a different file format is used because of a length constraint in the file names.*

4.4.3 Diagnostics Patches Description for ZeBu Server 2

Diagnostics patches for ZeBu Server 2 are typically installed in
<installation_directory>/etc/firmwares/ZSE/dt/V5.6.

4.4.3.1 Example of 2-slot Single Unit ZeBu Server 2 System With 5F/ICE Modules

XC6V-203_53_53.diag

It corresponds to your system configuration:

- XC6V → ZeBu Server 2 system
- 203 → 2: 2 slots, 0: single-unit, 3: unit version
- 53 → 5: 5F/ICE module, 3: module board version

4.4.3.2 Example of 5-slot Single Unit ZeBu Server 2 System With 9F/ICE Modules

XC6V-503_92_92_92_92_92.diag

It corresponds to your system configuration:

- XC6V → ZeBu Server 2 system
- 503 → 5:5 slots, 0: single-unit, 3: unit version
- 92 → 9: 9F/ICE module, 2: module board version

4.4.3.3 Example of Multi Unit ZeBu Server 2 System (with up to 5 units)

XC6V-C00-02-514_162_162_162_162_162-514_162_162_162_162.diag

It corresponds to your system configuration:

- XC6V → ZeBu Server 2 system
- C00-02 → <Cxx-yz>, where:
 - Cxx → Cable configuration label (C00 to C99)
 - y → Id of the unit where the hub is plugged (0: unit Id; default unit: U0)
 - z → Hub version (2: hub version)

NOTE: C00-00: *multi unit system in a 1-unit configuration (-00 → no hub).*

- 514 → 5: 5 slots, 1: multi unit, 4: unit version
- 162 → 16:17F module, 2: module board version

4.4.3.4 Example of Multi Unit ZeBu Server 2 System (with more than 5 units)

XC6V-C02-10U_84481.diag

It corresponds to your system configuration:

- XC6V → ZeBu Server 2 system
- C02 → Cable configuration label (C00 to C99)
- 10U → Number of units (06U to 10U)
- 84481 → Interconnection topology

Note

In the last example (configurations with more than 5 units), a different file format is used because of a length constraint in the file names.

4.4.4 Diagnostics Patches Description for ZeBu Server 3

Diagnostics patches for ZeBu Server 3 are typically installed in
<installation_directory>/etc/firmwares/ZSE/dt/V6.0.

4.4.4.1 Example of 2-slot Single Unit ZeBu Server 3 System With one 9F/ICE Module and one 9F Module

XC7V-203_92_81.diag

It corresponds to your system configuration:

- XC7V → ZeBu Server 3 system
- 203 → 2: 2 slots, 0: single-unit, 3: unit version
- 92 → 9: 9F/ICE module, 2: module board version
- 81 → 8: 9F module, 1: module board version

4.4.4.2 Example of 5-slot Single Unit ZeBu Server 3 System With 9F/ICE Modules

XC7V-503_92_92_92_92.diag

It corresponds to your system configuration:

- XC7V → ZeBu Server 3 system
- 503 → 5:5 slots, 0: single-unit, 1: unit version
- 92 → 9: 9F/ICE module, 2: module board version

4.4.4.3 Example of Multi Unit ZeBu Server 3 System (with up to 5 units)

XC7V-C00-02-514_81_81_81_81_81-514_81_81_81_81.diag

It corresponds to your system configuration:

- XC7V → ZeBu Server 3 system
- C00-02 → <Cxx-yy>, where:

- ❑ Cxx → Cable configuration label (C00 to C99)
- ❑ y → Id of the unit where the hub is plugged (0: unit Id; default unit: U0)
- ❑ z → Hub version (2: hub version)

NOTE: C00-00: *multi unit system in a 1-unit configuration (-00 → no hub).*

- 514 → 5: 5 slots, 1: multi unit, 4: unit version
- 81 → 8:9F module, 1: module board version

4.4.4.4 Example of Multi Unit ZeBu Server 3 System (with more than 5 units)

XC7V-C02-10U_84481.diag

It corresponds to your system configuration:

- XC7V → ZeBu Server 3 system
- C02 → Cable configuration label (C00 to C99)
- 10U → Number of units (06U to 10U)
- 84481 → Interconnection topology

Note

In the last example (configurations with more than 5 units), a different file format is used because of a length constraint in the file names.

4.5 Diagnostics Patches Description for ZeBu Server 4

Diagnostics patches for ZeBu Server 4 are typically installed in
<installation_directory>/etc/firmwares/ORION/dt/V1.0

4.5.1 Example Unit Configuration of ZeBu Server 4

4.5.1.1 XCVU-1U-12C Diagnostics

The diagnostics adapt to your system configuration. The 1U-12C diagnostics are compatible with a mono-unit system, with any number of modules from 1 to 4:

- XCVU: ZeBu Server 4 system
- 1U: Number of units
- 12C: Type of module

4.6 Post- Installation

The installation directory (/usr/share/zebu/O-2018.09/ in the previous example) contains the ZeBu tools and the installed Xilinx Place & Route software.

Diagnostics patches are to be installed after the initial installation of the ZeBu software release with the Synopsys Installer, as described in section [Example of Configuration Script For Xilinx Place & Route Software Installation](#).

The directory containing the installation files (where a temporary installation directory named snps_installer_temp_<username> was also created) may be deleted once the installation is complete.

At this point, you have an operational compilation chain for any Linux PC after configuring the end-user's environment as described in chapter [Configuring End-User's Environment](#) and creating the target hardware configuration file as described in chapter [Target Hardware Configuration File](#).

If the present PC is connected to the ZeBu system (host PC for emulation runtime), you should follow indications in chapter [Configuring End-User's Environment](#) to chapter [Initializing the ZeBu Server System](#) before proceeding with emulation.

Note

Once the ZeBu software is installed, it must not be copied to another directory in the network.

4.7 Installing Licenses for ZeBu

Using ZeBu requires the following FLEXnet licenses:

- For the ZeBu software license:
- The FLEXnet license daemon (lmgrd)
- The Synopsys vendor license daemon (snpslmd)
- The license file
- For the Xilinx ISE or Vivado Place & Route software license:
- The FLEXnet license daemon (lmgrd)
- The Xilinx vendor license daemon (xilinxd)
- The license file

For information about FLEXnet licensing tools, you can get documentation from the Flexera™ website at <http://www.flexerasoftware.com>. Following FLEXnet recommendations, it is recommended to launch the license manager without root privileges for security reasons.

4.7.1 Installation Process for Licenses

4.7.1.1 Installing ZeBu License Server

For more information on the ZeBu license server installation, see the [Downloading & Installing SCL](#) section of the [Synopsys Licensing QuickStart Guide](#), available on the Synopsys website.

4.7.1.2 Installing Xilinx License Server

You can install the FLEXnet license manager on a PC (with Linux operating system) or on a Sun SPARC (with Solaris operating system), as described in section [Network](#)

[Access to the ZeBu Software](#). The installation process is the same for both operating systems.

The Xilinx license vendor daemon (xilinxd) is available in the following directories of the ZeBu software release.

- For Xilinx ISE Place & Route:

- ☐ `<install_path> /ise/ISE_DS/ISE/bin/lin/xilinxd`
- ☐ `<install_path> /ise/ISE_DS/ISE/bin/lin64/xilinxd`
- ☐ `<install_path> /ise/ISE_DS/ISE/bin/sol/xilinxd`

- For Xilinx Vivado Place & Route:

- ☐ `<install_path>/vivado/bin/unwrapped/lnx64.0/xilinxd`

The typical name of the Xilinx license file is `Xilinx.lic`.

Each delivered license file is specific to one license server (host).

4.7.2 Modifying the License File

4.7.2.1 Modifying the ZeBu License File

For more information on the modification of the ZeBu license file, see the [Customizing the License Key File](#) section of the [Synopsys Licensing QuickStart Guide](#), available on the Synopsys website.

4.7.2.2 Modifying Xilinx License File

Once you have received the license files, they must be modified to match your IT configuration. The line `SERVER <hostname> <hostid> <portnumber>` must be modified with the following information:

- Change `<hostname>` to your server hostname (using the result of `uname -n` command on the license server)
- Change `<portnumber>` to an unused TCP port number (the port number must be higher than 1024 when the license manager is launched with user privileges as recommended by the FLEXnet user documentation)

Installing Licenses for ZeBu

- The `<hostid>` value must not be modified manually (if this value is modified, the license file will no longer be valid). If it does not match the result of `lmhostid` on the license server, you should contact Synopsys support.

It is recommended to copy the license files and the vendor daemons in the same directory as the license manager utilities. However, if the vendor daemons are not stored in the same directory as the license manager daemon (lmgrd), you should also modify the **VENDOR** line of the license file with the path to the vendor daemon:

```
VENDOR <vendor_daemon> <path_to_vendor_daemon>
```

Note

*If the vendor daemon is in the same directory as the license manager daemon, the **VENDOR** line must be present but the path information is ignored.*

Example:

Following is part of the initial content of the license file for the Xilinx software:

```
SERVER hostname1 12345678 2100  
USE_SERVER  
VENDOR xilinxd
```

After modification, it should look as follows:

```
SERVER myHost 12345678 1235  
USE_SERVER  
VENDOR xilinxd /local/licenses/zebu/xilinxd
```


4.7.3 Starting the License Server

4.7.3.1 Starting the ZeBu License Server

For more information on how to start the ZeBu license server, see [Starting the License Server](#) section of the [Synopsys Licensing QuickStart Guide](#), available on the Synopsys website.

4.7.3.2 Starting the Xilinx License Server

The appropriate command to start the Xilinx license server is:

```
$ <lm_path>/lmgrd -c <Xilinx_lic_file> -l <lm_logpath>/
<Xilinx_lic_logfile>
```

where:

- `<lm_path>` is the path to the license manager daemon
- `<Xilinx_lic_file>` is the path to the Xilinx license file
- `<lm_logpath>` is the path where you want the license log files to be stored

Once the `$LM_LICENSE_FILE` variable is set as described in section [Activating a User License](#), you can check that the license server is running correctly using the `lmstat -a` command (the host name is `myHost` and the port number is 1235):

```
lmstat - Copyright (c) 1989-2006 Macrovision Europe Ltd. and/or
Macrovision Corporation. All Rights Reserved.
Flexible License Manager status on Mon 11/3/2013 19:21
License server status: 2100@myHost
    License file(s) on myHost: /remote/license/data/Xilinx/
myHost_Xilinx.lic:
    myHost: license server UP (MASTER) v11.6
```

```
Vendor daemon status (on myHost):  
  xilinxd: UP v11.6  
Feature usage info:  
Users of Logic_Edition:  (Total of 20 licenses issued;  Total of 0  
licenses in use)  
Users of System_Edition:  (Total of 20 licenses issued;  Total of 0  
licenses in use)  
Users of ChipscopePro:  (Total of 40 licenses issued;  Total of 0  
licenses in use)  
Users of ISE:  (Total of 40 licenses issued;  Total of 0 licenses  
in use)  
Users of SDK:  (Total of 20 licenses issued;  Total of 0 licenses  
in use)  
Users of ChipScopePro_SIOTK:  (Total of 40 licenses issued;  Total  
of 0 licenses in use)  
Users of ISIM:  (Total of 40 licenses issued;  Total of 0 licenses  
in use)  
Users of XPS:  (Total of 20 licenses issued;  Total of 0 licenses  
in use)  
Users of Simulation:  (Total of 20 licenses issued;  Total of 0  
licenses in use)  
Users of SysGen:  (Total of 20 licenses issued;  Total of 0  
licenses in use)  
Users of HLS:  (Total of 20 licenses issued;  Total of 0 licenses  
in use)  
Users of PlanAhead:  (Total of 40 licenses issued;  Total of 0  
licenses in use)  
Users of Implementation:  (Total of 20 licenses issued;  Total of 0  
licenses in use)  
Users of Analyzer:  (Total of 20 licenses issued;  Total of 0  
licenses in use)  
Users of Synthesis:  (Total of 20 licenses issued;  Total of 0  
licenses in use)
```

4.7.4 Updating the ZeBu License File

If you need to update your ZeBu license file, see the **Procedure to Update a Synopsys License File** section in the [SCL Administration Guide](#) (available in the Documentation section of the [Synopsys Licensing QuickStart Guide](#)).

4.8 Downloading Required Interoperable Technologies

ZeBu Server requires the download of specific Synopsys interoperable technologies when using the features listed in the following sections.

The version and license features required for these Synopsys products are described in the *ZeBu Server Release Notes*.

4.8.1 Unified Compile Feature

To use the Unified Compile feature, you must download and install the supported release of VCS MX.

4.8.2 Debugging Features

To benefit from enhanced debugging with ZeBu Server, you must download and install the supported releases of these additional Synopsys products:

- Verdi³
- Siloti, available in Novas section in SolvNet

4.8.3 Co-Simulation with VCS

For co-simulation with VCS, you will be required to use VCS runtime licenses acquired independently.

5 Configuring End-User's Environment

The operations described in this section are required for every ZeBu user and consists of the following sub-sections:

- *Mandatory Environment Variables for ZeBu Software*
- *Activating a User License*
- *Declaring the Setup Directory for Emulation Runtime*
- *Runtime Settings for Relocation*

5.1 Mandatory Environment Variables for ZeBu Software


5.1.1 ZeBu Compilation and Runtime Software


Based on the shell type you use, you must source the `zebu_env.<shell>` file which is in the installation directory to automatically set the following mandatory environment variables for the ZeBu Server compilation and runtime software:

- `ZEBU_ROOT`
- `LD_LIBRARY_PATH`
- `PATH`
- `ZEBU_DRIVER_PATH`

Sourcing the `zebu_env.<shell>` file also sets required environment variables for Xilinx ISE and Vivado Place & Route software.

Note

 *It is not recommended to manually set mandatory environment variables for ZeBu.*

 *For other third-party tools, you should refer to your supplier's recommendations.*

5.1.2 Enhanced Compilation Feature for ZeBu Server 2 Hardware Only

Triton is an optional tool that enhances cells placement within the Xilinx Vivado Place & Route software, thus providing better compilation performance. This feature is only available for Xilinx Virtex-7 FPGAs of ZeBu Server 2 through a specific environment variable.

5.2 Activating a User License

5.2.1 Activating a ZeBu License

For more information on how to activate a ZeBu license, see the [Setting Up the User Environment to Access the Key File](#) section of the [Synopsys Licensing QuickStart Guide](#), available on the Synopsys website.

5.2.2 Activating a Xilinx License

You should update the `LM_LICENSE_FILE` environment variable for each end-user with appropriate information.

The syntax is the following:

- In bash shell:

```
$ export  
LM_LICENSE_FILE=<Xilinx_portnb>@<hostname>:$LM_LICENSE_FILE
```

- In C shell:

```
setenv LM_LICENSE_FILE <Xilinx_portnb>@<hostname>:$LM_LICENSE_FILE
```

Where `<...portnb>` and `<hostname>` depend on the content of the corresponding license files.

Instead of setting the `LM_LICENSE_FILE` environment variable, it is possible to use a specific environment variable for Xilinx licenses.

Example of syntaxes:

- In bash shell:

```
$ export  
XILINXD_LICENSE_FILE=<Xilinx_portnb>@<hostname>:$XILINXD_LICENSE_F  
ILE
```

■ In C shell:

```
setenv XILINXD_LICENSE_FILE  
<Xilinx_portnb>@<hostname>:$XILINXD_LICENSE_FILE
```

Further information about setting these environment variables can be found in FLEXnet documentation.

5.3 Declaring the Setup Directory for Emulation Runtime

Before proceeding with emulation, the setup directory must be declared by each user by setting the `ZEBU_SYSTEM_DIR` environment variable.

This variable is mandatory in the same way as `ZEBU_ROOT` is mandatory to each ZeBu Server system. Following is an example of a path setting for the ZeBu Server:

```
$ export ZEBU_SYSTEM_DIR=<my_setup_directory>
```

where `<my_setup_directory>` is the path declared in the setup file (`<my_setup>.zini`), as described in section [Declaring the Setup Directory in the zini file](#).

5.4 Runtime Settings for Relocation

In a multi user environment, a design that was compiled for relocation can be run on different subsets of FPGA modules of the ZeBu system. It is not possible to run a design on a subset of FPGA modules if the `use_module` command has not been declared during compilation.

Note

This section applies to ZeBu Server 1, ZeBu Server 2 and ZeBu Server 3. For ZeBu Server 4, see the dedicated section.

By default, the design runs on the subset of FPGA modules declared for compilation with `use_module` commands, generally starting from M0. To run the design on other modules of the same unit, users must set the `ZEBU_M0_PHYSICAL_LOCATION` environment variable before launching emulation runtime (note that 0 in M0 is

Runtime Settings for Relocation

character 'zero', not capital 'O'). This environment variable re-defines at runtime the new physical location of the first module logically assigned during compilation.

In case of a multi unit system, this environment variable has additional constraints: the user must be connected on the appropriate ZeBu Server unit, as shown in the example below. Specific recommendations for relocation with a multi unit/multi PCIe system are available in section [Runtime Settings for Relocation With a Multi Unit/Multi PCIe System](#).

Information for the possible physical locations of M0 after compilation is available in the `loc.xref` file located in the `zcui.work/zebu.work/tools/zPar/` directory. The modules authorized for declaration are the ones listed in the `$U0.M0.physicalLoc` line of the file.

Note

When the design has been compiled for a single module with a hardware configuration file for a 5S unit, it can be used indifferently on a 2S or 5S unit. When the design has been compiled for a single module with a hardware configuration file for a 2S unit, it can be used on a 2S or on modules M0 or M1 of a 5S unit.

When initializing the emulation, the module declared in the `ZEBU_M0_PHYSICAL_LOCATION` environment variable is checked for coherence with the information resulting from the compilation of the design and with the actual ZeBu system. **zServer** provides the following message to determine on which module the design is run:

```
-- ZeBu : zServer : The design will be relocated on physical Unit 1
Module 0.
```

Note

If the expected resources are currently used for another design, the emulation is queued by zServer (except if the `ZEBU_EXIT_IF_NO_RESOURCE_IS_AVAIL` environment variable has been previously set to "OK": the emulation stops instead of being queued).

Example:

For a design that uses 1 module compiled on a 2-unit system with 3 modules, the following line appears in the `loc.xref` file:

```
$U0.M0.physicalLoc=" U0.M0 U0.M1 U1.M0 " ;
```

This means that logical module M0 of the design can be assigned to `U0.M0`, `U0.M1` or `U1.M0`.

When emulation is launched from a PC connected to U0, user can declare either of:

```
$ export ZEBU_M0_PHYSICAL_LOCATION=M0  
or  
$ export ZEBU_M0_PHYSICAL_LOCATION=M1
```

When emulation is launched from a PC connected to U1, user can declare:

```
$ export ZEBU_M0_PHYSICAL_LOCATION=M0
```

5.4.1 Runtime Settings for Relocation With a Multi Unit/Multi PCIe System

When using a multi unit system with several PCIe boards in the host PCs, the `ZEBU_U0_PHYSICAL_LOCATION` environment variable can be set to declare the ZeBu Server physical unit, which is connected as logical U0.

If this environment variable is not set, logical U0 is the unit connected to the first PCIe board detected by the BIOS of the host PC. For more information on the BIOS detection of PCIe boards, see section [PCIe Boards Detection Order](#).

Example:

For a design that uses two modules, compiled on a 3-unit system, the following line appears in the `loc.xref` file:

```
$U0.M0.physicalLoc=" U0.M0 U1.M0 U2.M0 " ;  
$U0.M1.physicalLoc=" U1.M0 U2.M0 U0.M0 " ;
```

This means that the design can be assigned to M0 modules of U0 and U1, or to M0

Runtime Settings for Relocation

modules of U1 and U2 or to M0 modules of U2 and U0.

Using a single PC with three PCIe boards, the design is launched by default on U0 and U1, equivalent to:

```
$ export ZEBU_U0_PHYSICAL_LOCATION=U0
```

To launch the design on U1 and U2, declare:

```
$ export ZEBU_U0_PHYSICAL_LOCATION=U1
```

To launch the design on U2 and U0, declare:

```
$ export ZEBU_U0_PHYSICAL_LOCATION=U2
```

If another host PC with only one PCIe board is connected to U0, U1 or U2, there is no need to declare `ZEBU_U0_PHYSICAL_LOCATION` (there is only one possible location in such a case: the unit connected to the particular PC).

5.4.2 Runtime Settings for Relocation on ZeBu Server 4

When using a ZeBu Server 4, to run the design on other modules of the same unit, you must set the `ZEBU_PHYSICAL_LOCATION` environment variable before launching emulation runtime. This environment variable defines the new physical location of the first module logically assigned during compilation. The location of the first module of each unit must be specified using the `ZEBU_PHYSICAL_LOCATION` environment variable as follows:

```
export ZEBU_PHYSICAL_LOCATION="U2.M0,U3.M0"
```

When a design uses only one half-module, use the following syntax:

```
export ZEBU_PHYSICAL_LOCATION="U2.HM1"
```

When initializing the emulation, the module declared in the

ZEBU_PHYSICAL_LOCATION environment variable is checked for consistency with the information resulting from the compilation of the design and with the actual ZeBu system. **zServer** provides the following message to indicate on which module the design is run:

```
-- ZeBu : zServer : Remapping required:  
-- ZeBu : zServer :    U0.M0 -> U0.M1
```

Note

If the expected resources are not available, the emulation stops and is not queued.

Example

To run your design on modules starting with M2, use the following:

```
$ export ZEBU_PHYSICAL_LOCATION=U0.M2
```

If the design fits on one half-module, use the following:

```
$ export ZEBU_PHYSICAL_LOCATION=U0.HM3
```

6 Initializing the Host PCs

This chapter provides information about initialization of PCIe boards on host PC and consists of the following sub-sections:

- [Prerequisites](#)
- [Initializing PCIe Boards on Host PC Running ZeBu 2017.03 or Newer Releases](#)
- [Initializing PCIe Boards on Host PC Running ZeBu 2017.03 and Older Releases \(2015.03 or newer\)](#)
- [Initializing the PCIe Boards for ZeBu Releases Older Than 2017.03 \(2015.03 or newer\)](#)
- [Initializing the PCIe Boards Automatically When Booting Linux](#)
- [PCIe Boards Detection Order](#)
- [zInstall Log Files](#)
- [Updating PCIe Boards Firmware](#)

For the ZeBu Server system, the initialization of PCIe boards on the host PC requires loading a specific ZeBu Device Driver or a ZeBu loadable kernel module. The loading of a specific ZeBu Device Driver or a ZeBu loadable kernel module is similar to loading a driver for any PCIe peripheral running with Linux operating system.

This ZeBu Device Driver is part of the ZeBu drivers package and is available on Solvnet. This package must be installed on the host PC using the Synopsys installer. For more information about the installation, see *Synopsys Installer Instruction Manual*. This is the only possibility for ZeBu Server 4.

The ZeBu loadable kernel module is specific to a ZeBu release and is provided with the corresponding ZeBu release. It is not possible to use the ZeBu loadable kernel module with ZeBu Server 4.

The process to install the ZeBu Device Driver or the ZeBu Kernel Module depends on the versions of the ZeBu releases used on the same host PC. The following are three different scenarios.

- Host PC runs only ZeBu 2017.03 or newer releases

The latest version of the driver must be installed on the host PC, as described in [Initializing PCIe Boards on Host PC Running ZeBu 2017.03 or Newer Releases](#). This is the only applicable scenario for ZeBu Server 4.

- Host PC runs both ZeBu 2017.03 and older releases (2015.03 or newer)
The initialization step of all releases is performed using the Device Driver Switching solution, with its **zSwitchDriver** executable and its associated `zInstallLauncher.sh` script, as described in [Initializing PCIe Boards on Host PC Running ZeBu 2017.03 and Older Releases \(2015.03 or newer\)](#). This scenario is not applicable to ZeBu Server 4.
- Host PC runs only releases older than ZeBu 2017.03
The initialization step of all releases is performed using the Device Driver Switching solution, with its **zSwitchDriver** executable and its associated `zInstallLauncher.sh` script, as described in [Initializing the PCIe Boards for ZeBu Releases Older Than 2017.03 \(2015.03 or newer\)](#). This scenario is not applicable to ZeBu Server 4.

For more information about the Device Driver Switching solution, see chapter [Appendix B: ZeBu Device Driver Switching Solution](#).

In addition to the initialization, the PCIe boards in the host PCs must be occasionally updated when installing a new software version or an intermediate patch. For more information, see [Updating PCIe Boards Firmware](#).

6.1 Prerequisites

- The ZeBu Device Drivers package must be downloaded and installed. It is recommended to always use the latest ZeBu Device Drivers package.
The instructions to download the latest ZeBu Device Drivers package are available on SolvNet.
 - ❑ [Instructions to Download the ZeBu Device Drivers Package for ZeBu Server 4](#)
 - ❑ [Instructions to Download the ZeBu Device Drivers' Package for ZeBu Server 1, 2, or 3](#)
- The ZeBu loadable kernel module (`zKernel`) is specific to a given ZeBu release. The specific requirements for the Linux kernel that can be used with ZeBu are detailed in chapter [Linux Operating System Requirements](#). It is provided in the ZeBu release package.
- If you have several host PCs in your configuration, the initialization step described in [Initializing PCIe Boards on Host PC Running ZeBu 2017.03 or Newer Releases](#) and [Initializing the PCIe Boards for ZeBu Releases Older Than 2017.03 \(2015.03 or newer\)](#) must be performed on each host PC.

6.2 Initializing PCIe Boards on Host PC Running ZeBu 2017.03 or Newer Releases

Note

- 📄 *All ZeBu emulation runs must be stopped during this initialization process.*
- 📄 *All commands described in this section must be launched with root permissions.*
- 📄 *This is the only applicable method for ZeBu Server 4*

6.2.1 Installing Driver on Host PC

To install the driver on the host PC, execute the following script available in the installed package:

```
<driver_package_path>/drivers/zebu_zs/zDriverInstall install
```

Where:

`driver_package_path` corresponds to the path where the ZeBu driver package is installed using the Synopsys installer.

Note

The execution of this script is required only when preparing the PC for the ZeBu installation for the first time.

The ZeBu Device Driver, `zebu_zs`, is permanently added as a device driver on the runtime PC. The ZeBu Device Driver is re-launched automatically when the PC is restarted.

The driver installation script creates the following directories on the host PC:

- `/zebu`
- `/zebu/queue`
- `/zebu/zUtils_logs`

Note

ZeBu software applications require these directories.

6.2.2 Removing Driver From Host PC

To remove the driver from the host PC, execute the following script available in the installed package:

```
<driver_package_path>/drivers/zebu_zs/zDriverInstall remove
```

Where:

- `driver_package_path` corresponds to the path where the ZeBu drivers package is installed using the Synopsys installer.

The device driver is permanently removed from the host PC.

6.3 Initializing PCIe Boards on Host PC Running ZeBu 2017.03 and Older Releases (2015.03 or newer)

In addition to the ZeBu Device Driver, the Zebu drivers package contains the Device Driver Switching solution, with its **zSwitchDriver** executable and its associated `zInstallLauncher.sh` script. For more information, see chapter [Appendix B: ZeBu Device Driver Switching Solution](#). The initialization step of all releases is performed using the Device Driver Switching solution.

Note

📄 You must launch the `zInstallLauncher.sh` script for each version.

📄 All ZeBu emulation runs must be stopped during this initialization process.

📄 All commands described in this section must be launched with root permissions. It is recommended to always use the latest version of **zSwitchDriver**.

📄 This section is not applicable to ZeBu Server 4.

6.3.1 Preparing the PC for ZeBu Releases Installation

To clean your workstation, execute the following command:

```
<driver_package_path>/zSwitchDriver/zInstallLauncher.sh -clean
```

Note

The execution of this script is required only when preparing the PC for ZeBu installation for the first time.

6.3.2 Initializing the PCIe Boards for ZeBu 2017.03 and Newer Releases


To install the ZeBu Device Driver that is used by O-2018.09 and newer releases, use the following syntax:


```
<driver_package_path>/zSwitchDriver/zInstallLauncher.sh
-start -r <zebu_release_path>
[-driverPath <driver_package_path>/drivers/zebu_zs]
[-driverModulePath <path_to_driver_module_file>]
```

Where:

- `driver_package_path` corresponds to the path where the ZeBu drivers package is installed using the Synopsys installer.
- `zebu_release_path` corresponds to the `$ZEBU_ROOT` of the ZeBu release.

Note

 `-driverPath` is optional. It is required, if the package hierarchy is modified (directories moved from the ZeBu drivers package) to indicate the path of the `zebu_zs` driver.

 `-driverModulePath` is optional, and indicates the path to the directory containing the pre-compiled driver (`zebu_zs.ko`)

The system is initialized and ready.

6.3.3 Initializing the PCIe Boards for ZeBu Releases Older Than 2017.03 (2015.03 or newer)

To install the ZeBu loadable kernel module that is used by releases prior to O-2018.09, use the following syntax:

```
<driver_package_path>/zSwitchDriver/zInstallLauncher.sh  
-start -r <zebu_release_path>
```

Where:

- `driver_package_path` corresponds to the path where the ZeBu drivers package is installed using the Synopsys installer.
- `zebu_release_path` corresponds to the `$ZEBU_ROOT` of the ZeBu release.

Note

`zInstallLauncher.sh` must be executed multiple times, once for each release prior to O-2018.09 release.

The system is initialized and ready.

By default, `zInstallLauncher.sh` searches for the `zKernel` object file (`zKernel.o`) in the following directories:

- `<zebu_release_path>/bin` (equivalent to `$ZEBU_ROOT/bin`)
- `/zebu`

If the `zKernel` object file cannot be found in these directories, `zInstallLauncher.sh` automatically compiles `zKernel` from the source file provided in the ZeBu software package.

6.4 Initializing the PCIe Boards Automatically When Booting Linux

While booting Linux, you can launch `zInstallLauncher.sh` by adding the following line in the `/etc/inittab` file (just before the `ttys'` initialization):

```
zInstallLauncher.sh -start -r <zebu_release_1_path>
zInstallLauncher.sh -start -r <zebu_release_2_path>
zInstallLauncher.sh -start -r <zebu_release_N_path>
```

where:

- `zebu_release_n_path` corresponds to the `$ZEBU_ROOT` of the ZeBu releases.

You must specify the full path of the software release (equivalent to `$ZEBU_ROOT`), since the environment variables are set after the `inittab` file is interpreted.

6.4.1 Pre-compiling the ZeBu Device Driver on Host PC

Note

You are recommended to follow the steps provided in the [Installing Driver on Host PC](#) section. The solution described in this section allows you to separate compilation and installation of the Linux Device Driver in different steps.

To compile the device driver on the host PC, perform the following steps:

1. Navigate to the `<driver_package_path>/drivers/zebu_zs/src` directory.
2. Execute the following commands:

```
setenv PATH /usr/bin:$PATH (optional to set the proper gcc version)
<driver_package_path>/drivers/zebu_zs/zDriverInstall compile
```

The `zebu_zs.<kernel_version>.ko` file specific to the version of Linux kernel is created in the `<driver_package_path>/drivers/zebu_zs/module` directory.

The install command automatically uses the appropriate pre-compiled ZeBu Device Driver.

Note

The execution of this script is required only when the PC is prepared for the ZeBu installation for the first time.

6.5 PCIe Boards Detection Order

The units must be connected to the PCIe boards in ascending order.

There may be gaps in the unit connection order. That is, you may connect U0, U2 and U4 on three boards of your host PC. However, the connection of the boards must be always in ascending order. that is, you cannot connect the boards in the following order: U0, U2 and U1.

The order for the addressing the PCIe boards in the host PC is defined by the BIOS.

Note

The indications printed on the motherboard of the host PC should not be taken into account.

To detect the order of the PCIe boards, perform the following steps:

1. Install your PCIe boards.
2. Launch the following command to stop a running **zSwitchDriver** and clean the environment:

```
zInstallLauncher.sh -clean
```

3. Set the environment variable `ZEBU_DEBUG_BOARD_IN_PC_MASK` with a mask for each PCIe board as follows:
 - ☐ Bit 0 of the mask is for first PCIe board (index 0)
 - ☐ Bit 1 of the mask is for second PCIe board (index 1), and so on

Examples:

- ❑ If you want to detect the first PCIe board, you must set the ZEBU_DEBUG_BOARD_IN_PC_MASK variable to 0x1.
 - ❑ If you want to detect all the five PCIe boards, you must set the ZEBU_DEBUG_BOARD_IN_PC_MASK variable to 0x1f (which is equivalent to have no mask at all).
4. Launch the following command to display all information about the unmasked PCIe boards:

```
zInstallLauncher.sh -start -r <zebu_release_path>
```

5. Use the indexes to connect the units in ascending order.

NOTE: Check the serial number that is physically indicated on each board to identify it.

When the `zUtils -getlabel` command is launched, specific information about the order is displayed and may be useful to fix the order of units correctly.

- When the detection order is correct, the following information is displayed:

```
-- ZeBu : zUtils : PCIe 0 slot 05:00.0, board 0950 is detected
(remap 128MB).
-- ZeBu : zUtils : PCIe 1 slot 8a:00.0, board 0920 is detected
(remap 128MB).
-- ZeBu : zUtils : PCIe 2 slot 90:00.0, board 0940 is detected
(remap 128MB).
-- ZeBu : zUtils : PCIe 3 slot 87:00.0, board 0910 is detected
(remap 128MB).
-- ZeBu : zUtils : PCIe 4 slot 84:00.0, board 0900 is detected
(remap 128MB).
```

- When the detection order is incorrect, an error message is generated indicating how PCIe boards must be connected to units:

```
-- ZeBu : zUtils : ERROR : LUI2058E : Bad unit detection order.
-- ZeBu : zUtils : ERROR : PCIe 0 board is normally connected to
unit 0, PCIe 1 board to unit 1, and so on.
-- ZeBu : zUtils : ERROR : You can connect PCIe 0 board to another
unit if, and only if, the other PCIe boards are connected in an
orderly way.
-- ZeBu : zUtils : ERROR : For instance:
-- ZeBu : zUtils : ERROR : - if you connect PCIe 0 to unit 1, then
PCIe 1 must be connected to unit 2, PCIe 2 to unit 3, etc.
-- ZeBu : zUtils : ERROR : - if you connect PCIe 0 to unit 2, then
PCIe 1 must be connected to unit 3, PCIe 2 to unit 4, etc.

-- ZeBu : zUtils : ERROR : The PCIe 0 board 0950 (slot 05:00.0), is
currently connected to unit 3.
-- ZeBu : zUtils : ERROR : The PCIe 1 board 0920 (slot 8a:00.0), is
currently connected to unit 2.
-- ZeBu : zUtils : ERROR : The PCIe 2 board 0940 (slot 90:00.0), is
currently connected to unit 1.
-- ZeBu : zUtils : ERROR : The PCIe 3 board 0910 (slot 87:00.0), is
currently connected to unit 4.
-- ZeBu : zUtils : ERROR : The PCIe 4 board 0900 (slot 84:00.0), is
currently connected to unit 0.
-- ZeBu : zUtils : ERROR : You should swap the cables connected to
the PCIe boards to have the proper order.
```

6.6 zInstall Log Files

`zInstallLauncher.sh` uses **zInstall** to generate log files that are present in the following paths:

- `/zebu/zInstall.<i>.log` (log files)
`<i>=0-4` (0 for the most recent file; older files are shifted for each **zInstall**)
- `/var/log/messages` (system file), or `/bin/dmesg` (executable, which reads the last lines of `/var/log/messages`).

When the ZeBu Server unit is initialized, **zSwitchDriver** becomes a daemon and continues to run in the background.

The `/zebu/zInstall.<i>.log` files are common to all connections, regardless of the ZeBu release.

6.6.1 Connection/Disconnection Logs

A history file for all connection and disconnection events (for **zInstall**, **zServer**, **zUtils** and customer testbenches) is stored as `/zebu/zebu_<year_month_day>.log` on a daily basis.

It is available for the last 15 days of use and each tool is listed as:

```
#boardId #userName #tool #pid at #date - #time : #STATUS or
command line
```

Where tool can be **zInstall**, **zServer**, **zUtils**, or name of a customer testbench.

Example:

Connection log progresses when user1 uses a ZeBu Server 2 unit (board ID = 00a0) with a C++ testbench (`testbench_prd`).

1. The testbench launches **zServer**:

```
user1 testbench_prd 9733 at <date> - <time> : ** OPEN *****
user1 testbench_prd 9733 at <date> - <time> : run_zebu/
testbench_prd
user1 zServer          9747 at <date> - <time> : ** OPEN *****
user1 zServer          9747 at <date> - <time> : zServer -zebu.work
.../zcui.work/zebu.work
```

2. **zServer** requests the connection with the kernel:

```
user1 zServer          9747 at <date> - <time> : ** CONNECT *****
```

NOTE: *If the board is being used by another process, zServer waits for the board to be available (this can last several minutes).*

3. **zServer** is connected to the board and the user testbench can use the board (which starts its initialization process):

```
00a0 user1 zServer          9747 at <date> - <time> : ** LOAD
***** using U2.M0
00a0 user1 testbench_prd 9733 at <date> - <time> : ** CONNECT
*****
```

4. The testbench starts running as soon as the connection with **zServer** is established. When the testbench ends, it requests disconnection from **zServer** to close the session:

```
user1 zServer          9747 at <date> - <time> : ** CLOSE *****
user1 zServer          9747 at <date> - <time> : code 0 : Server
closed correctly.
```

5. The testbench terminates correctly:

```
00a0 user1 testbench_prd 9733 at <date> - <time> : ** CLOSE
*****
00a0 user1 testbench_prd 9733 at <date> - <time> : code 0 :
Simulation finished
```


6.7 Updating PCIe Boards Firmware

The PCIe boards in the host PCs sometimes must be updated when installing a new software version or an intermediate patch. A dedicated tool, **zUpdate**, is available for updating the PCIe boards.

When the PCIe update is required, see the corresponding ZeBu Server Release Notes or patch note.

6.7.1 Updating Only One PCIe Board in Host PC

In this case, use the following command:

```
$ zUpdate
```

You must restart the host PC once this update is complete.

6.7.2 Updating Several PCIe Boards in Host PC

To update several PCIe boards in the host PC, an index can be declared for **zUpdate** using the following command:

```
$ zUpdate [<PCIeIndex>]
```

Where <PCIeIndex> can be any integer between 0 and 4 (If <PCIeIndex> is not set, only the board with index 0 is updated).

Note

This command must be launched as many times as the number of available PCIe boards.

You must restart the host PC once this update is complete (After all the **zUpdate** commands are run).

7 ZeBu Server System Setup

This chapter provides information about **zSetupSystem** tool for completing the ZeBu Server setup process and consists of the following sub-sections:

- [*zSetupSystem Tool*](#)
- [*Choosing the Setup Directory*](#)
- [*Preparing the Setup File*](#)
- [*Launching zSetupSystem*](#)
- [*Content of the Setup Directory*](#)

For the first installation of the ZeBu Server system and after any hardware modification (if a unit or module has been added, changed, or removed or if the cables interconnecting units have been moved), a setup process must be launched for system calibration and tests. A dedicated tool, **zSetupSystem**, supports the complete setup process, as described in section [*zSetupSystem Tool*](#).

The data resulting from diagnostics and calibration process is stored by **zSetupSystem** in a dedicated directory for the entire ZeBu Server system, known as the setup directory. To support multi unit and multi user features all the PCs connected to the ZeBu Server system must have Read/Write accesses to this directory.

After the setup process, the setup directory includes the results of diagnostics and calibration processes as well as the hardware configuration file for ZeBu compilation.

Before proceeding with emulation runtime, this setup directory must be declared with the `ZEBU_SYSTEM_DIR` environment variable.

7.1 zSetupSystem Tool

zSetupSystem is a dedicated tool to launch system calibration and diagnostics, test the memories, and generate the target hardware configuration file for compilation.

zSetupSystem integrates the following steps:

- Initialization, detection, verification, and calibration of the ZeBu Server system with **zUtils**:
 - ❑ Initializes the ZeBu Server system.
 - ❑ Detects the full configuration of all modules in the system.
 - ❑ Generates a `config.dff` configuration file with module IDs.
 - ❑ Calibrates all inter-FPGA connections at all predefined frequencies and tests all memories.
 - ❑ Generates a `status_se.tcl` file containing the test results.
- Generation of the target hardware configuration file for compilation with **zConfig**:
 - ❑ Using the `status_se.tcl` file generated in the previous step), generates the target hardware configuration file, `zse_configuration.tcl`, and the FLLP history file (frequency-limited LVDS pairs) for the current ZeBu Server system, as described in section [Target Hardware Configuration File](#).
- Generation of the host topology file for use by the Resource Management API:
 - ❑ This is mandatory for ZeBu Server 4 and optional for the other ZeBu platforms.
 - ❑ This file describes the connection between the host PCs and the ZeBu system.
 - ❑ The Resource Management API uses it to determine the possible placements of a compiled design.
- Storage of system information:
 - ❑ A compressed archive, `<config_directory>.tar.bz2`, is stored in the directory from where **zSetupSystem** was launched (for example `my_setup_dir.tar.bz2`). This archive is intended for tracking purposes and should be sent to your Synopsys representative if requested.

7.2 Choosing the Setup Directory

7.2.1 One Directory for Each System

It is mandatory to create one setup directory for each ZeBu system.

7.2.1.1 One Directory for All Releases

It is recommended to use the same setup directory (\$ZEBU_SYSTEM_DIR) for all software versions.

However, this directory must be generated using the latest major ZeBu release.

7.2.1.2 One Directory for All PCs

When several PCs are connected to the same ZeBu Server 2 system, a symbolic link can be created to the setup directory of the ZeBu Server 2 system with the name of each PC, as in the following example for PC0:

```
ln -s /usr/share/zebu/setup/203_92_81 /usr/share/zebu/setup/  
my_ZSE_on_PC0
```

With such a link, setting the \$ZEBU_SYSTEM_DIR environment variable for emulation runtime using PC0 is easier because only the name of the PC is required from user:

```
$ export ZEBU_SYSTEM_DIR=/usr/share/zebu/setup/my_ZSE_on_PC0
```

7.2.1.3 Naming Rules

There is no naming rule for the setup directory; any path name can be used. However, it is recommended to choose an understandable name, when several non-identical systems may be available.

Example:

Using the ZeBu Server 3 system type, as done by Synopsys for diagnostics patch names, provides a non-ambiguous naming convention (see section [Diagnostics Patches Description for ZeBu Server 3](#)). An index can be added to differentiate systems with identical configurations:

- If the <sys_config> information in the diagnostics patch name for your system is 203_92_81, you can set the following value:

```
$ZEBU_SYSTEM_DIR="/usr/share/zebu/setup/203_92_81";
```

- If you have several systems with the same <sys_config> information, for example 203_92_81, you can use A and B indexes to differentiate the 2 paths:

- ☐ For system A:

```
$ZEBU_SYSTEM_DIR="/usr/share/zebu/setup/203_92_81_A";
```

- ☐ For system B:

```
$ZEBU_SYSTEM_DIR="/usr/share/zebu/setup/203_92_81_B";
```

7.3 Preparing the Setup File

7.3.1 Setup File Template

The setup file template supplied with this release, `setup_template.zini`, is available after software installation in the `$ZEBU_ROOT/etc/configurations/` directory and described in section [Appendix C: setup_template.zini](#). This file must be copied locally on the host PC then modified manually for your actual configuration (the template contains helpful comments for tag modifications).

7.3.2 Declaring the Setup Directory in the `zini` file

Declaring the path to the setup directory is mandatory in the setup file:

```
$ZEBU_SYSTEM_DIR="my_setup_dir" ;
```

In the simplest case where you installed only one diagnostics patch corresponding to the configuration of your ZeBu Server system, this is the only mandatory declaration.

7.3.3 Declaring the diagnostics patches (if several are installed)

If several diagnostics patches are installed in your release, you must declare which patch is used for the present system initialization. For that purpose, the `$patchFile` line must be modified with the path to the appropriate patch.

Example: For a 5 slot multi unit backplane with five 9F/ICE modules:

```
$patchFile = "/zebu/release/etc/firmwares/ZSE/dt/V6.0/XC7V-C00-00-514_91_91_91_91_91.diag" ;
```

For details on diagnostics patches naming syntax, see section [Diagnostics Patches Overview](#).

7.3.4 Managing Calibration History

By default, **zSetupSystem** uses the history file, which is automatically stored in the setup directory.

To ignore the history, the `$noHistory` variable can be set to 1 in the setup file:

```
$noHistory = 1 ;
```

In the template file, this line is present with a comment mark for an easy modification.

When a new setup directory is declared in the setup file, no history file is considered.

7.3.5 Memory Tests

7.3.5.1 Testing the Memories

By default, **zSetupSystem** tests all the memories at the end of setup, after the calibration of the ZeBu Server system. This is equivalent to:

```
$ zUtils -mem all_mem
```

Information on faulty/available memories is logged in two different files:

- `status_se.tcl` (in a section called: ZeBu-SE_memories)
- `memories.dff` (in a section called: memories status)

7.3.5.2 Example of `memories.dff` for a ZeBu Server 1 system:

```
$U0_M0_F16_DDR2_0 = "available";  
$U0_M0_F16_DDR2_1 = "available";  
$U0_M0_FS_DDR2_0 = "available";  
$U0_M1_F16_DDR2_0 = "available";  
$U0_M1_F16_DDR2_1 = "faulty";  
$U0_M1_FS_DDR2_0 = "available";  
$U0_M2_F16_DDR2_0 = "available";  
$U0_M2_F16_DDR2_1 = "available";  
$U0_M2_FS_DDR2_0 = "available";
```


7.3.5.3 Example of `memories.dff` for a ZeBu Server 2 system:

```
$U0_M0_F16_DDR2_0 = "available";
$U0_M0_F16_DDR2_1 = "available";
$U0_M0_FS_DDR2_0 = "available";
$U0_M2_F16_DDR2_0 = "available";
$U0_M2_F16_DDR2_1 = "available";
$U0_M2_FS_DDR2_0 = "available";
$U1_M2_F16_DDR2_0 = "available";
$U1_M2_F16_DDR2_1 = "available";
$U1_M2_FS_DDR2_0 = "available";

$U2_M2_F16_DDR2_1 = "available";
$U2_M2_FS_DDR2_0 = "available";
$U3_M2_F16_DDR2_0 = "available";
$U3_M2_F16_DDR2_1 = "available";
$U3_M2_FS_DDR2_0 = "available";
$U4_M2_F16_DDR2_0 = "available";
$U4_M2_F16_DDR2_1 = "available";
$U4_M2_FS_DDR2_0 = "available";
$U5_M2_F16_DDR2_0 = "available";
$U5_M2_F16_DDR2_1 = "available";
$U5_M2_FS_DDR2_0 = "available";
```

7.3.5.4 Example of `memories.dff` for a ZeBu Server 3 system:

```
$U0_M0_F00_DDR_0 = "available";
$U0_M0_F00_DDR_1 = "available";
$U0_M0_F01_DDR_0 = "available";
$U0_M0_F01_DDR_1 = "available";
$U0_M0_F06_DDR_0 = "available";
$U0_M0_F06_DDR_1 = "available";
$U0_M0_F07_DDR_0 = "available";
$U0_M0_F07_DDR_1 = "available";
$U0_M0_F08_DDR_1 = "available";
$U0_M0_FS_DDR_0 = "available";
$U0_M0_FS_DDR_1 = "available";
$U0_M1_F00_DDR_0 = "available";

$U0_M1_F00_DDR_1 = "available";
$U0_M1_F01_DDR_0 = "available";
$U0_M1_F01_DDR_1 = "available";
$U0_M1_F06_DDR_0 = "available";
$U0_M1_F06_DDR_1 = "available";
$U0_M1_F07_DDR_0 = "available";
$U0_M1_F07_DDR_1 = "available";
$U0_M1_F08_DDR_0 = "available";
$U0_M1_F08_DDR_1 = "available";
$U0_M1_FS_DDR_0 = "available";
$U0_M1_FS_DDR_1 = "available";
```

Runtime errors caused by faulty memories can be found in section [Runtime Errors Caused by Faulty Memories](#).

7.3.5.5 Example of `memories.dff` for a ZeBu Server 4 System

```
# ===== Memories status =====

$U0_HM0_F01_DDR_0 "available"
$U0_HM0_F01_DDR_1 "available"
$U0_HM0_F02_DDR_0 "available"
$U0_HM0_F02_DDR_1 "available"
$U0_HM0_F03_DDR_0 "available"
$U0_HM0_F04_DDR_0 "available"
$U0_HM0_F04_DDR_1 "available"
$U0_HM0_FC_DDR_0 "available"

$U0_HM1_F01_DDR_0 "available"
$U0_HM1_F01_DDR_1 "available"
$U0_HM1_F02_DDR_0 "available"
$U0_HM1_F02_DDR_1 "available"
$U0_HM1_F03_DDR_0 "available"
$U0_HM1_F04_DDR_0 "available"
$U0_HM1_F04_DDR_1 "available"
$U0_HM1_FC_DDR_0 "available"
```

7.3.5.6 Skipping Memory Tests

To skip the memory tests carried out at the end of setup, the `setup_template.zini` file must be modified before launching **zSetupSystem**. The following line should be uncommented:

```
#$noMemTest = 1;
```

7.3.6 Reserving Memory for SRAM-Trace (Static-Probes)

Note

This section applies to ZeBu Server 1 and ZeBu Server 2 hardware only.

The amount of memory reserved in each module for SRAM-trace applies to the entire ZeBu Server system during initialization, whatever the configuration (for different types of modules in the system), and applies to all users. The default memory capacity for SRAM-trace is 256 MBytes.

To modify the reserved memory capacity, the `setup_template.zini` file must be edited before launching **zSetupSystem**. The following line should be modified based on your ZeBu Server system:

```
$TRACE_SIZE = <size>;
```

Where `<size>` is a size in 32-bit words and must be set in hexadecimal.

- For ZeBu Server 1
 - ❑ For systems with 8C/ICE modules: `<size>` maximum value is `0x20000000` (equivalent to 2 GBytes)
 - ❑ For systems with only 4C or 16C modules: `<size>` maximum value is `0x40000000` (equivalent to 4 GBytes)
 - ❑ Default value is `0x04000000` (equivalent to 256 MBytes)
- For ZeBu Server 2
 - ❑ `<size>` maximum value for all modules is `0x40000000` (equivalent to 4 GBytes)
 - ❑ Default value is `0x04000000` (equivalent to 256 MBytes)

Note

In case of a ZeBu Server 1 system containing heterogeneous modules with different physical memory sizes (for example, 8C/ICE and 16C modules with 2 GBytes and 4 GBytes), if user declares a 3 GBytes SRAM-trace size then the complete memory will be reserved for SRAM-trace in the 8C/ICE module.

The memory reserved for SRAM-trace does not impact the available resource for the

design or testbench memory, since it is connected to one control FPGA in each module.

7.3.6.1 Balancing SRAM-Trace (Static-Probes) Capacity and Bitstream Cache

ZeBu Server features a cache system that allows the ZeBu Server system to load faster when the same design is used repeatedly:

- **On ZeBu Server 1 and ZeBu Server 2 hardware**, this cache is implemented using the physical memory and used by SRAM-trace: if not completely reserved for SRAM-trace, it is used to store (cache) the bitstream files of design and RTB FPGAs.

If the complete memory is reserved for SRAM-trace, there is no available space for the cache and all the bitstream files must be downloaded from the host PC to the ZeBu Server system each time the emulation starts. The tables in sections [Bitstream Cache Capacities for ZeBu Server 1](#) and [Bitstream Cache Capacities for ZeBu Server 2](#) show the number of designs that can be stored in the cache based on the memory capacity reserved for SRAM-trace.

- **On ZeBu Server 3 hardware**, 8 GBytes of physical memory are dedicated with the following mechanisms:
 - ❑ 4 GBytes of memory resources are reserved in each module to store the bitstream files of FPGAs
 - ❑ 4 GBytes of memory resources are available for SRAM-trace in each module; however, during a runtime job, only the SRAM-trace on the U0.M0 module can be used, that is, 4 GBytes total for the DUT memory.

This cache works as a circular buffer to keep only the most recent bitstream files. If the bitstream files of a design are not present in the cache, they are downloaded from the host PC to the cache before the ZeBu Server system is loaded.

The content of the cache is erased at system initialization with **zSetupSystem** or with **zUtils** (when it used with **-initSystem** or **-DT** options).

In case of an error when loading FPGAs from the bitstream cache, the bitstream files are automatically downloaded from the host PC without using the cache mechanism.

Bitstream Cache Capacities for ZeBu Server 1

TABLE 12 Bitstream Cache Capacity (default SRAM-trace size = 256 MBytes)

Module Type	Physical Memory Size	SRAM-Trace Size	Max. # of Designs in Bitstream Cache
4C	4 GBytes	256 MBytes	69
8C	2 GBytes	256 MBytes	26
16C	4 GBytes	256 MBytes	35

TABLE 13 Bitstream Cache Capacity (SRAM-trace size = 0)

Module Type	Physical Memory Size	SRAM-Trace Size	Max. # of Designs in Bitstream Cache
4C	4 GBytes	0	73
8C	2 GBytes	0	29
16C	4 GBytes	0	37

TABLE 14 Bitstream Cache Capacity (max. SRAM-trace size for 8C modules = 2 GBytes)

Module Type	Physical Memory Size	SRAM-Trace Size	Max. # of Designs in Bitstream Cache
4C	4 GBytes	2 GBytes	36
8C	2 GBytes	2 GBytes	0
16C	4 GBytes	2 GBytes	18

TABLE 15 Bitstream Cache Capacity (max. SRAM-trace size for 4C/16C modules = 4 GBytes)

Module Type	Physical Memory Size	SRAM-Trace Size	Max. # of Designs in Bitstream Cache
4C	4 GBytes	4 GBytes	0
16C	4 GBytes	4 GBytes	0

Bitstream Cache Capacities for ZeBu Server 2

TABLE 16 Bitstream Cache Capacity (default SRAM-trace size = 256 MBytes)

Module Type	Physical Memory Size	SRAM-Trace Size	Max. # of Designs in Bitstream Cache
5F/ICE	4 GBytes	256 Mbytes	29
9F/ICE	4 GBytes	256 Mbytes	17
17F	4 GBytes	256 Mbytes	9

TABLE 17 Bitstream Cache Capacity (SRAM-trace size = 0)

Module Type	Physical Memory Size	SRAM-Trace Size	Max. # of Designs in Bitstream Cache
5F/ICE	4 GBytes	0	30
9F/ICE	4 GBytes	0	18
17F	4 GBytes	0	10

TABLE 18 Bitstream Cache Capacity (SRAM-trace size = 2 GBytes)

Module Type	Physical Memory Size	SRAM-Trace Size	Max. # of Designs in Bitstream Cache
5F/ICE	4 GBytes	2 GBytes	15

TABLE 18 Bitstream Cache Capacity (SRAM-trace size = 2 GBytes)

9F/ICE	4 GBytes	2 GBytes	8
17F	4 GBytes	2 GBytes	5

When the trace size is set to its maximum (4 GBytes), the bitstream cache capacity is 0 for all the FPGA modules and the designs are reloaded from the host PC each time also in case of repeated use.

Bitstream Cache Capacities for ZeBu Server 3

TABLE 19 Bitstream Cache Capacity for ZeBu Server 3

Module Type	Physical Memory Size	SRAM-Trace Size	Max. # of Designs in Bitstream Cache
All module types	4 GB	2 GB	15

Bitstream Cache Capacities for ZeBu Server 4

TABLE 20 Bitstream Cache Capacity for ZeBu Server 4

Module Type	Physical Memory Size	SRAM-Trace Size	Max. # of Designs in Bitstream Cache
12C	2 x 8 GB	Not Available	13

7.4 Launching zSetupSystem

zSetupSystem is controlled by a setup file, `<my_setup>.zini`, which includes information such as the path to the setup directory and the firmware version of the ZeBu Server system. A template for this setup file, `setup_template.zini`, is available in the `$ZEBU_ROOT/etc/configurations` directory and described in [Preparing the Setup File](#).

Launching zSetupSystem

The command line to launch the setup process is:

```
$ zSetupSystem <my_setup>.zini -platform [ orionzs1|zs2|zs3|zs4 ]
[-topology]
```

Note

*The license environment must be setup before launching **zSetupSystem**.*

7.4.1 Multiple Host PCs

■ Single unit configuration:

In this configuration, you can launch **zSetupSystem** from any host PC.

■ Multi unit configuration:

In this configuration, you MUST proceed in the following order:

- a. Connect the hub to connector C4 of unit U0.
- b. Launch **zSetupSystem** from any host PC connected to U0.

Note

ZeBu Server 1, ZeBu Server 2, ZeBu Server 3

*In a multi user system configuration (single or multi unit), if one of the host PCs connected to any unit is switched OFF, you may no longer can communicate with the units from any PC connected to the system. In most cases, communication resumes as soon as ALL the PCs are switched back ON. If not, you must power OFF/ON all units then launch **zSetupSystem** (described in this section) or **zUtils -initSystem** (described in section [Initializing the System With zUtils -initSystem](#))*

7.4.2 Creating the Host Topology

zSetupSystem displays a list of questions about the connection between the host PCs and the ZeBu system and you must always answer them for ZeBu Server 4. However, it is optional (when the `-topology` option is used) to answer the questions on other ZeBu platforms.

The following is an example of **zSetupSystem** dialog.

```

-- ZeBu : zUtils :
=====

-- ZeBu : zUtils : ==== Creation of (ZEBU_SYSTEM_DIR)/
host_topology.xml ====
-- ZeBu : zUtils :
=====

Registration of a new host:
+ Enter the name of the host pc: zebu_host05
| Registration of a new Pci connection for 'zebu_host05':
| Pci slot: 0
| Pci connector: 0
| Unit ID: 0
| Unit connector: 0

+ Do you want to register another Pci connection for 'zebu_host05'?
(y/n) n

Do you want to register another host machine? (y/n) n

-- ZeBu : zUtils : ==== (ZEBU_SYSTEM_DIR)/host_topology.xml created
=====

```

Where

- `Pci slot` is the number of the PCIe slot inside the host PC.
- `Pci connector` is the number of the connector on the ZeBu PCIe board.
- `Unit ID` is the ID of the physical unit linked by this ZeBu PCIe board to the host PC.
- `Unit connector` is the ID (0-4) of the connector (C0-C4) on the ZeBu unit used for this connection.

7.5 Content of the Setup Directory

7.5.1 Setup Directory Files

The following files are generated in the setup directory (or updated if they already exist):

TABLE 21 Files in the Setup Directory

File	Description
config.dff	Complete architecture of the detected ZeBu Server system.
memories.dff	Memories status.
status_se.tcl	System status for the design compilation process (input file for zConfig).
<my_setup>.zini	Copy of the setup file, with the same name as the zSetupSystem input file.
host_topology.xml	Description of the connection between the host PCs and the ZeBu system.

7.5.2 Sub-Directories in the Setup Directory

The setup directory also includes the following sub-directories, which are generated by **zSetupSystem** or updated if they already exist:

TABLE 22 Subdirectories in the Setup Directory

Directory	Description
dt	Contains all input files for zUtils –calibration and zUtils –DT user tests.
calibration	Contains all calibration files (except for ZeBu Server 4).
history	Contains Frequency-Limited LVDS Pairs (FLLP) history files.
config	Output directory generated by zConfig .

TABLE 22 Subdirectories in the Setup Directory

logs/info	Contains information about all the connections and connection attempts to a ZeBu system. These daily log files are named with the current date as follows: sw_connectionInfo_YYYY_MM_DD.log (see section sw_connectionInfo_YYYY_MM_DD.log) connections_YYYY_MM_DD.xml (see Section connections_YYYY_MM_DD.xml)
logs/srdCpu	Contains access reservation logs to hardware resources (does not apply to ZeBu Server 4) Format: srdCpu_<yyyy_mm_dd>.log
logs/connection	Contains temporary connection files
logs/run	Contains runtime log files. For more information, see ZeBu Runtime Log Files .
logs/ temperature	Contains temperature log files. For more information, see Temperature Log File . Format: temperature_<yyyy_mm_dd>.log

7.5.3 Information Log Files

7.5.3.1 sw_connectionInfo_YYYY_MM_DD.log

The content of sw_connectionInfo_YYYY_MM_DD.log files in the logs/info directory looks like the following example:

```
(1) <date> - <time> : <pid> started (<hostname> - <username> - pid
<pid>)
(2) <date> - <time> : <pid> get connection
(3) <date> - <time> : <pid> reserve : unit <unit_id> <info_reg>
(modules <Mx My>)
(4) <date> - <time> : <pid> free
(5) <date> - <time> : <pid> OK
```

Content of the Setup Directory

Each line corresponds to a step below:

- (1) The process starts on the host PC.
- (2) The process accesses the ZeBu system.
- (3) The process runs on the ZeBu system using the unit and module listed. There are as many such lines as units for the process.
`<info_reg>` is a binary value which can be used by Synopsys for support purposes.
- (4) (5) The process is terminated.

This file can optionally include the size of a process design (i.e. the number of FPGAs, of LUTs and of registers). This optional information requires a specific license feature (`ZSE_resource_designSize`) and is present if the `ZEBU_USE_RESOURCE_LICENSES` environment variable is set to ON before launching emulation runtime. This information is displayed after line (3) in the log file as follows:

```
<date> - <time> : <pid> resource : nbFpga <nb_FPGA>, nbLut
<nb_LUT>, nbReg <nb_registers>
```

When this size information is added in the `sw_connectionInfo_YYYY_MM_DD.log` file, it is also available in the `srdCpu_<date>.log` file in the `$ZEBU_SYSTEM_DIR/logs/srdCpu/` directory.

7.5.3.2 connections_YYYY_MM_DD.xml

The `connections_YYYY_MM_DD.xml` files in the `logs/info` directory contain information regarding connections to a specific ZeBu system for a specific ZeBu release. XML files are more accessible and contain machine-readable context information that allows data to be read and used by other tools.

The content of `connections_YYYY_MM_DD.xml` files looks like the following example:

```
<?xml version="1.0" encoding="ISO-88858-1"?>
<connectionInfo date="20160418">
  <open id="58221839" date="20160418-15:23:51">
    <pc>host</pc>
    <pid>22559</pid>
    <user>username</user>
    <group>usergroup</group>
    <release>/usr/share/zebu/O-2018.09/</release>
    <systemDir>/auto/ZeBu/host</systemDir>
    <currentDir>/auto/design/</currentDir>
    <project>projectname</project>
    <team>teamname</team>
  </open>
  <reserve id="58221839" date="20160418-15:23:51"/>
  <load id="58221839" date="20160418-15:23:51">
    <nbUnits>1</nbUnits>
    <unit index="0">
      <modulesList nbModules="2">M0 M1</modulesList>
    </unit>
  </load>
  <close id="58221839" date="20160418-15:24:07">
    <pc>host</pc>
    <pid>22559</pid>
    <user>username</user>
    <group>usergroup</group>
```

```

<release>/usr/share/zebu/0-2018.09/</release>
<systemDir>/auto/ZeBu/host</systemDir>
<currentDir>/auto/design</currentDir>
<project>projectname</project>
<team>teamname</team>
<nbUnits>1</nbUnits>
<unit index="0">
  <modulesList nbModules="2">M0 M1</modulesList>
</unit>
<openPendingDuration>00:00:00</openPendingDuration>
<reservePendingDuration>00:00:02</reservePendingDuration>
  <runningDuration>00:00:14</runningDuration>
<driverClk>8.333 MHz 27,934,600</driverClk>
  </close>
</connectionInfo>

```

where:

- <open>: The process starts on the host PC.
- <reserve>: The process accesses the ZeBu system.
- <load>: The process runs on the ZeBu system using the listed unit(s) and module(s). There are as many lines as units for this process.
- <close>: The process is terminated.

7.5.3.3 Temperature Log File

A temperature log file is generated in \$ZEBU_SYSTEM_DIR/logs/temperature.

This file, temperature_<yyyy_mm_dd>.log, includes three types of information:

■ System event information:

```
zebu_event <run_Id> <time> <event> <type>
```

where:

- ❑ <run_Id>: User ID for runtime
- ❑ <time>: date/time (format: "Weekday DD MM YYYY - HH:MM:SS")
- ❑ <event>: load or close
- ❑ <type>: TB (user testbench), IS (system init), of DT (diagnostics)

■ Temperature information:

❑ Modules temperatures

```
zebu_temp <run_Id> <time> <module_phys_Id> <module_type>  
<module_mfg_id> <temps>
```

where:

- ◆ <module_phys_Id>: module's physical Id (format: "Ui Mj")
- ◆ <module_type>: module's board type, for example, zse_4c_lx330_v3
- ◆ <module_mfg_Id>: module's manufacturing Id, i.e., 0x0090
- ◆ <temps>: FPGA temperatures in °C (format: {Fx=TT Fy=TT ...})

❑ PCies temperatures

```
pcie_temp <hostname> S<n> <pcie_mfg_Id> {S<n>.PE.Fx=<temp>}  
  
pcie_max_temp <hostname> {S0.PE.Fx=<max temp> .. Sn.PE.Fx=<max  
temp>}  
  
pcie_min_temp <hostname> {S0.PE.Fx=<min temp> .. Sn.PE.Fx=<min  
temp>}
```

Content of the Setup Directory

where:

- ◆ <hostname>: Host PC name of the PCIe board(s)
- ◆ S<n>: Slot number of the PCIe in the host PC
- ◆ <pcie_mfg_Id>: PCIe's manufacturing ID
- ◆ S<n>.PE.FX: FPGA on the S<n> PCIe
- ◆ <[min|max] temp>: temperature in degree Celsius

■ Fan speed information:

```
zebu_fan <run_Id> <time> <unit_phys_Id> <speeds>
```

where:

- <unit_Id>: unit's physical Id (format: Ux)
- <fan_speeds>: fan speed coefficients for <unit_Id> (format: {SP SP})

7.5.3.4 Example of Temperature log file for ZeBu Server 2:

```
version "1.0"
```

Note: The following references are all physical.

```
zebu_event          32 "<date> - 11:06:15" load      TB

zebu_temp           32 "<date> - 11:06:15" "U0 M0"
zse_xc7v_8c_2000_v1 0x26f2 {FC=99 FS=98 IF=99 F00=99 F01=99 F02=99
F03=99 F04=99 F05=99 F06=99 F07=99 F08=99}

zebu_temp           32 "<date> - 11:06:15" "U0 M1"
zse_xc7v_8c_2000_v1 0x2c72 {FC=99 FS=98 IF=99 F00=99 F01=99 F02=99
F03=99 F04=99 F05=99 F06=99 F07=99 F08=99}

zebu_fan            32 "<date> - 11:06:15" U0 {0 0}
zebu_max_temp       32 "<date> - 11:06:15" U0 {80 80}
pcie_temp           32 "<date> - 11:06:15" pc_03  S0  0x0740
{S0.PE.FX=50}

pcie_min_temp       32 "<date> - 11:06:15" pc_03  {S0.PE.Fx=49 }
pcie_max_temp       32 "<date> - 11:06:15" pc_03  {S0.PE.Fx=54 }

zebu_temp           32 "<date> - 11:06:15" "U0 M0"
zse_xc7v_8c_2000_v1 0x26f2 {FC=99 FS=98 IF=99 F00=99 F01=99 F02=99
F03=99 F04=99 F05=99 F06=99 F07=99 F08=99}

zebu_temp           32 "<date> - 11:06:15" "U0 M1"
zse_xc7v_8c_2000_v1 0x2c72 {FC=99 FS=98 IF=99 F00=99 F01=99 F02=99
F03=99 F04=99 F05=99 F06=99 F07=99 F08=99}

zebu_fan            32 "<date> - 11:06:15" U0 {0 0}
zebu_max_temp       32 "<date> - 11:06:15" U0 {80 80}
pcie_temp           32 "<date> - 11:06:15" pc_03  S0  0x0740
{S0.PE.FX=50}
```

```
pcie_min_temp      32 "<date> - 11:06:15" pc_03 {S0.PE.Fx=49 }
pcie_max_temp      32 "<date> - 11:06:15" PC_03 {S0.PE.Fx=54 }
zebu_event         32 "<date> - 11:07:03" close      TB
```

7.5.4 ZeBu Runtime Log Files

Besides **zUtils** and **zInstall** log files which are saved in /zebu local subdirectories, the runtime logs (that is, testbenches, **zServer** and **zRun**) are by default stored in the \$ZEBU_SYSTEM_DIR/logs/run directory as listed in [Table 22](#).

The log files are named as follows:

```
#processName.#hostName.#userName.#date.log
```

Symbolic links are also available to the five most recent logs in this directory (#processName.#hostName.#index.log).

7.5.4.1 Modifying the Storage Directory and Names of Runtime Log Files

You can modify:

- The default name of the runtime logs directory (\$ZEBU_SYSTEM_DIR/logs/run/) with the *ZEBU_LOGS_DIRECTORY* environment variable
- The default runtime log file names with the *ZEBU_LOGS_NAME_SUFFIX* environment variable

With both environment variables set, the path of the log file is as follows:

```
$ZEBU_LOGS_DIRECTORY/#processName.$ZEBU_LOGS_NAME_SUFFIX.log
```

Note

When the `ZEBU_LOGS_NAME_SUFFIX` variable is set, the date in the name of the file is removed.

When the `ZEBU_LOGS_DIRECTORY` and the `ZEBU_LOGS_NAME_SUFFIX` environment variables are set, the `designFeatures.help` file is also named like the other runtime log files: `$ZEBU_LOGS_DIRECTORY/designFeatures.$ZEBU_LOGS_NAME_SUFFIX.help`

7.5.4.2 Multi Day Runs

When a runtime job lasts several days, all the information about this job is registered into several runtime log files across the log directories. These log files have a different date, which makes all the information available as expected in the archives.

Example:

A runtime job starts on Day1 and ends on Day3. Runtime logs for this job will be created:

- On Day1, at job start (that is, `sw_connectionInfo_YYYY_MM_Day1.log`)
- On Day2, if information must be logged (that is, `sw_connectionInfo_YYYY_MM_Day1.log`)
- On Day3, at job end (that is, `sw_connectionInfo_YYYY_MM_Day3.log`)

7.5.4.3 Compressing and Deleting old Runtime Log Files

The oldest files are never deleted by the ZeBu software; they are compressed and it is the user's responsibility to decide whether to retain them or not.

The storage of all runtime log files can be controlled using the following environment variables:

- `ZEBU_LOGS_MAX_AGE`: Maximum age of the compressed log files (default: 15 days)
- `ZEBU_LOGS_GZIP_CMD`: Command used for file compression (default: `gzip`)
- `ZEBU_LOGS_GZIP_OPTIONS`: `gzip` command option (default: `-f`)

7.5.4.4 Links to Logs in the Run Directory

In the directory where the ZeBu run is performed, links point to the five most recent files. —

8 Initializing the ZeBu Server System

This chapter provides information about initialization of ZeBu Server with the **zSetupSystem** tool and consists of the following sub-section(s):

- *Initializing the System With zUtils -initSystem*

8.1 Initializing the System With **zUtils** - **initSystem**

You must initialize the ZeBu Server system with **zUtils** each time a power OFF/ON occurred on at least one unit in the system (without any modification of the interconnection between units).

Run the following command from any PC connected to unit U0:

```
$ export ZEBU_SYSTEM_DIR=<my_system_dir>
$ zUtils -initSystem
```

This loads the backplane(s) and the modules' system FPGAs.

Note

*You must initialize the ZeBu Server system with **zSetupSystem** each time a configuration change occurs in your ZeBu Server system. Modification of the interconnection between units is considered as a modification of the hardware configuration of the system. In such a case, do not use **zUtils** -initSystem. To consider this change and avoid runtime failure and possible hardware damages, you must use **zSetupSystem**, as described in [ZeBu Server System Setup](#).*

9 Target Hardware Configuration File

This chapter provides information about how to create the target hardware configuration file of a ZeBu Server system and consists of the following sub-sections:

- [Generating a Configuration File for One ZeBu Server System](#)
- [Generating a Configuration File for Several ZeBu Server Systems](#)
- [Generating a Hardware Configuration File for Relocation](#)
- [Generating a Default Configuration File](#)
- [Using a Sample Configuration File](#)

The target hardware configuration file is a Tcl script describing the modules plugged in each unit of a ZeBu Server system. This description file is mandatory for the ZeBu compiler and is declared in the compilation project.

Three different possibilities exist:

- **Generated hardware configuration file for the actual system:**

This file is either generated during the setup of the system, as described in chapter [ZeBu Server System Setup](#) or generated manually with **zConfig** as described in sections [Generating a Configuration File for One ZeBu Server System](#) to [Default Configuration File for a Specific Interconnection Topology](#) for specific configurations.

It takes into account the diagnostics of the actual ZeBu Server system and can be used to compile when targeting runtime emulation.

- **Sample configuration file:**

Some sample configuration files, listed in section [Using a Sample Configuration File](#), are delivered with the software release for evaluation purposes: you can compile for most of the available ZeBu Server system configurations without the actual hardware installed.

These sample files can ONLY be used for compilation.

- **Generated default configuration file:**

This file is generated manually with **zConfig** to compile designs and estimate the performance in a single or multi unit environment without the actual ZeBu Server units (the diagnostics are not considered), as described in section [Default Configuration File for a Specific Interconnection Topology](#).

It is interesting to compile when no sample hardware configuration file matches the system you want to evaluate.

These generated files can ONLY be used for compilation.

If you must compile for several ZeBu Server systems, **zConfig** has some options to generate a hardware configuration file which is compatible with your systems in the following cases:

- Several ZeBu Server systems with the same units and modules, in terms of type and position (see section [Hardware Configuration File for Non-Identical ZeBu Server Systems](#)).
- Several non-identical ZeBu Server systems (see section [Generating a Hardware Configuration File for Relocation](#)):

The generated hardware configuration file will support only the modules which are similar in all the systems.

The hardware configuration file of a ZeBu Server system is always generated along with a frequency-limited LVDS pairs (FLLP) history file, `status_U<N>_<XXXX>.pin` (<N> is the ID of a ZeBu Server unit and <XXXX> is the serial number of the same unit in hexadecimal format).

9.1 Generating a Configuration File for One ZeBu Server System

The hardware configuration file for one ZeBu Server system is automatically generated during the setup of the system with **zSetupSystem**, as described in chapter [ZeBu Server System Setup](#).

In such a case, the generated file considers the diagnostics of the ZeBu Server system.

The hardware configuration file is named `zse_configuration.tcl` and is automatically stored in the setup directory (`$ZEBU_SYSTEM_DIR`). This information will be an input for the ZeBu compiler, in **zCui** or in the Unified Compile UTF file.

9.2 Generating a Configuration File for Several ZeBu Server Systems

When targeting to use the compiled design with several ZeBu Server systems, it is necessary to manually create a single hardware configuration file with **zConfig** tool.

Generating such a file requires that each targeted ZeBu Server system has been set up previously with **zSetupSystem**, as described in chapter [ZeBu Server System Setup](#), since the `status_se.tcl` files are required.

9.2.1 Hardware Configuration File for Identical ZeBu Server Systems

If you intend to use ZeBu Server systems, which are identical in terms of units and modules (type and position in the system), **zConfig** must be used with the `status_se.tcl` file generated in the setup directory for each system (the command has been split on several lines for legibility):

```
$ zConfig -cfg <$ZEBU_SYSTEM_DIR_1>/status_se.tcl
          -cfg <$ZEBU_SYSTEM_DIR_2>/status_se.tcl
          -cfg <$ZEBU_SYSTEM_DIR_n>/status_se.tcl
          -o <output_path>
```

Systems

The resulting `zse_configuration.tcl` file makes compilation results compatible between all <n> ZeBu Server systems.

Note

zConfig also generates an FLLP file for each ZeBu Server unit present in each of the <n> ZeBu Server systems.

9.2.2 Hardware Configuration File for Non-Identical ZeBu Server Systems

For ZeBu Server systems that are NOT identical in terms of units and modules, if you use the same syntax as in the previous section, **zConfig** displays an error indicating that the merging of your systems is not possible.

In this case, use the `-mi` option (module intersection) to generate a configuration file which will be common to all your ZeBu Server systems:

```
$ zConfig -mi
    -cfg <$ZEBU_SYSTEM_DIR_1>/status_se.tcl
    -cfg <$ZEBU_SYSTEM_DIR_2>/status_se.tcl
    -cfg <$ZEBU_SYSTEM_DIR_n>/status_se.tcl
    -o <output_path>
```

Note

When the `-mi` option is used, zConfig keeps in `zse_configuration.tcl` only the units and modules which are identical in all ZeBu Server systems.

9.3 Generating a Hardware Configuration File for Relocation

In a multi user environment, relocation is possible only if the hardware configuration file considers simultaneously the diagnostics of all the FPGA modules in all systems, all the detected FLLPs.

For that purpose, **zConfig** can be launched with the `-merge` option:

```
$ zConfig -merge -cfg <path_0>/status_se.tcl -cfg <path_1>/  
status_se.tcl -o my_config
```

Note

The `-merge` option can be placed anywhere in the command line.

9.4 Generating a Default Configuration File

To compile designs and estimate the performance in a single or multi unit environment without any diagnostics file, that is, without the actual ZeBu Server units, you can generate a default configuration file with **zConfig**.

Two options can be used for that purpose:

- `-default`: use this option to generate a default configuration file with the list of modules in the ZeBu Server system you want to compile for.
- `-nbfpga`: use this option to generate a default configuration file with the number of FPGAs in the ZeBu Server system you want to compile for.

In a multi unit environment, it may be useful to generate a default configuration file which matches a specific interconnection topology between the units. This is applicable for the specific topology which provides a scalable system not requiring the modification of existing cables.

9.4.1 Default Configuration File Using a List of Modules

To generate a default hardware configuration file with the list of modules in the ZeBu Server system you want to compile for:

```
$ zConfig -default <system_config> [-mu] -o <output_path>
```

where:

- <output_path>: directory where zse_configuration.tcl is generated.
- <system_config>: description of the ZeBu Server system:
 - 2-slot single unit system: ab
 - 5-slot single unit system: abcde
 - Multi unit system (5-slot only): abcde_fghij_klmno_pqrst_uvwxy
 - -mu: optional toggle to create a multi unit based file when only one 5-slot system is declared.

Each of the above characters has one the following values:

TABLE 23 ZeBu Server 1 Syntax Rules

Character in Syntax Above	Value	Type of Module
a through y	0	No module
	4	4C
	8	8C
	I, i	ICE (8C)
	F, f	16C
	L, l	Loopback

TABLE 24 ZeBu Server 2 Syntax Rules

Character in Syntax Above	Value	Type of Module
a through y	0	No module
	4	5F/ICE
	8	9F/ICE
	F, f	17F

TABLE 25 ZeBu Server 3 Syntax Rules

Character in Syntax Above	Value	Type of Module
a through y	0	No module
	8	9F
	i	9F/ICE

TABLE 26 ZeBu Server 4 Syntax Rules

Character in Syntax Above	Value	Type of Module
a through y	0	No module
	M	12F
	H	Hub

9.4.1.1 Example Common to All ZeBu Server Hardware

- 2-slot single-unit system with an ICE module in M0 and no module in M1:

```
$ zConfig -default i0 -o path_to_myconfig
```

9.4.1.2 ZeBu Server 1 Examples

■ 5-slot ZeBu Server 1 system:

- To create a default configuration file for a 5-slot unit with an ICE module in M0, no module in M1 and 16C modules in M2, M3 and M4:

```
$ zConfig -zs1 -default i0FFF -o path_to_myconfig
```

- To create a default configuration file for the same 5-slot unit but for a multi unit environment:

```
$ zConfig -zs1 -mu -default i0FFF -o path_to_myconfig
```

■ Multi unit ZeBu Server 1 system (4 units in these examples):

```
$ zConfig -zs1 -default i0444_88888_f4i04_00FFF -o  
path_to_myconfig
```

where: 00FFF indicates that there is no module in U3.M0 and U3.M1.

```
$ zConfig -zs1 -default i0444_88888_f4i04_1LFFF -o  
path_to_myconfig
```

where: 1LFFF indicates that U3.M0 and U3.M1 are loopback modules.

9.4.1.3 ZeBu Server 2 Examples

■ 5-slot ZeBu Server 2 system:

- To create a default configuration file for a 5-slot unit with a 9F/ICE module in M0, no module in M1 and 17F modules in M2, M3 and M4:

```
$ zConfig -zs2 -default 80FFF -o path_to_myconfig
```


Generating a Default Configuration File

- ❑ To create a default configuration file for the same 5-slot unit but for a multi unit environment:

```
$ zConfig -zs2 -mu -default 80FFF -o path_to_myconfig
```

- Multi unit ZeBu Server 2 system (four units in these examples):

```
$ zConfig -zs2 -default i0444_88888_f4804_00FFF -o  
path_to_myconfig
```

where: 00FFF indicates that there is no module in U3.M0 and U3.M1.

9.4.1.4 ZeBu Server 3 Examples

- 5-slot system:

- ❑ To create a default configuration file for a 5-slot unit with a 9F/ICE module in M0, no module in M1 and 9F modules in M2, M3 and M4:

```
$ zConfig -zs3 -default i0888 -o path_to_myconfig
```

- ❑ To create a default configuration file for the same 5-slot unit but for a multi unit environment:

```
$ zConfig -zs3 -mu -default i0888 -o path_to_myconfig
```

- Multi unit system (4 units in these examples):

```
$ zConfig -zs3 -default i0888_iiii_i8i08_00888 -o path_to_myconfig
```

where: 00888 indicates that there is no module in U3.M0 and U3.M1.

9.4.1.5 ZeBu Server 4 Examples

To create a default configuration file with 4 FPGA modules:

```
$ zConfig -zs4 -default MMMM -o path_to_myconfig
```

9.4.2 Default Configuration File Using an FPGA Count

To generate a default hardware configuration file with the number of FPGAs in the ZeBu Server system you want to compile for:

```
$ zConfig -nbfpga <N> -o <output_path>
```

Where <N> is the number of FPGAs which will be generated (from 1 to 800).

Examples:

- Default configuration file for 80 FPGAs:

```
$ zConfig -nbfpga 80 -o path_to_myconfig
```

This generates a configuration file for 1 unit with five 17F modules (FFFFF).

- Default configuration file for 81 FPGAs:

```
$ zConfig -nbfpga 81 -o path_to_myconfig
```

This generates a configuration file for a 2-unit system:

- ❑ First unit (U0) with 5 x 17f modules (FFFFF)
- ❑ Second unit (U1) with 1 x 5F/ICE module in U1.M0 (40000)

9.4.3 Default Configuration File for a Specific Interconnection Topology

A default configuration file can be generated for multi unit systems with a specific interconnection topology. This is applicable for the specific topology which provides a scalable system not requiring the modification of existing cables.

To generate such a configuration file, **zConfig -topo** must be used simultaneously with either the **-default** option or the **-nbfpga** option:

```
$ zConfig -topo 84481-3 -default <system_config> -o <output_path>
```

or:

```
$ zConfig -topo 84481-3 -nbfpga <N> -o <output_path>
```

where:

- `<system_config>` describes the ZeBu Server system with same syntax as the description in section [Default Configuration File Using a List of Modules](#)).
- `<N>` is the number of FPGAs of the ZeBu Server system (from 4 to 800 to match all possible single unit or multi unit systems).
- `<output_path>`: directory where `zse_configuration.tcl` is generated.

In the present software version, only one additional interconnection topology (84481-3) is available which matches a scalable configuration. Synopsys can only update the list of supported values.

9.4.3.1 zConfig for ZeBu Server 4

To create a configuration for a single unit of four modules:

```
zConfig -zs4 -default MMMM -o cfg_single_unit
```

To create a configuration for multi unit:

```
zConfig -zs4 -default MMMM_MMMM_MMMM_HH -cabling <tone-file-wiring-3U> -o cfg_3unit
```

Note

You must provide your own wiring for multi unit configurations.

A sample cabling file is present in `${ZEBU_ROOT}/etc/sys/ORION/configs/orion_cabling*`.

Example of sample cabling command

```
zConfig -zs4 -default MMMM_MMMM_HH -cabling ${ZEBU_ROOT}/etc/sys/ORION/configs/orion_cabling_16U_8H.tcl -o <output_dir>
```

9.5 Using a Sample Configuration File

Some sample target hardware configuration files are available in the ZeBu package (after installation, they can be found in the `$ZEBU_ROOT/etc/configurations` directory). These files can ONLY be used for compilation.

Note

For runtime, you MUST generate a specific target hardware configuration file for your actual ZeBu Server system(s) before compilation (see sections [Generating a Configuration File for Several ZeBu Server Systems](#) and [Generating a Default Configuration File](#)).

9.5.1 ZeBu Server 1 Sample Configuration Files

TABLE 27 Available Sample Configuration Files

ZeBu Server 1 System	FPGA Modules	# of FPGAs	Name of Sample Configuration File
2-slot single-unit	1x4C	4	sample_ZSE_2S_4C_LX330.tcl
"	2x4C	8	sample_ZSE_2S_4C_4C_LX330.tcl
"	1x16C	16	sample_ZSE_2S_16C_LX330.tcl
"	2x16C	32	sample_ZSE_2S_16C_16C_LX330.tcl
"	1xICE*	8	sample_ZSE_2S_ICE_LX330.tcl
"	1xICE+ 1x4C*	12	sample_ZSE_2S_ICE_4C_LX330.tcl
"	1xICE+ 1x8C*	16	sample_ZSE_2S_ICE_8C_LX330.tcl
5-slot single-unit	1x16C	16	sample_ZSE_5S_16C_LX330.tcl

Using a Sample Configuration File

TABLE 27 Available Sample Configuration Files

ZeBu Server 1 System	FPGA Modules	# of FPGAs	Name of Sample Configuration File
"	2x16C	32	sample_ZSE_5S_16C_16C_LX330.tcl
"	3x16C	48	sample_ZSE_5S_16C_16C_16C_LX330.tcl
"	4x16C	64	sample_ZSE_5S_16C_16C_16C_16C_LX330.tcl
"	5x16C	80	sample_ZSE_5S_16C_16C_16C_16C_16C_LX330.tcl
"	1x16C + 1x8C*	24	sample_ZSE_5S_16C_8C_LX330.tcl
"	2x16C + 1x8C*	40	sample_ZSE_5S_16C_16C_8C_LX330.tcl
"	3x16C + 1x8C*	56	sample_ZSE_5S_16C_16C_16C_8C_LX330.tcl
"	4x16C + 1x8C*	72	sample_ZSE_5S_16C_16C_16C_16C_8C_LX330.tcl
"	1xICE+ 1x16C*	24	sample_ZSE_5S_ICE_16C_LX330.tcl
"	1xICE+ 4x8C*	40	sample_ZSE_5S_ICE_8C_8C_8C_8C_LX330.tcl
"	1xICE+ 4x16C*	72	sample_ZSE_5S_ICE_16C_16C_16C_16C_LX330.tcl
5-slot multi unit	1x16C	16	sample_ZSE_5SM_16C_LX330.tcl
"	2x16C	32	sample_ZSE_5SM_16C_16C_LX330.tcl
"	3x16C	48	sample_ZSE_5SM_16C_16C_16C_LX330.tcl
"	4x16C	64	sample_ZSE_5SM_16C_16C_16C_16C_LX330.tcl

TABLE 27 Available Sample Configuration Files

ZeBu Server 1 System	FPGA Modules	# of FPGAs	Name of Sample Configuration File
"	5x16C	80	sample_ZSE_5SM_16C_16C_16C_16C_LX330.tcl
"	1x16C + 1x8C*	24	sample_ZSE_5SM_16C_8C_LX330.tcl
"	2x16C + 1x8C*	40	sample_ZSE_5SM_16C_16C_8C_LX330.tcl
"	3x16C + 1x8C*	56	sample_ZSE_5SM_16C_16C_16C_8C_LX330.tcl
"	4x16C + 1x8C*	72	sample_ZSE_5SM_16C_16C_16C_16C_8C_LX330.tcl
"	1xICE+ 1x16C*	24	sample_ZSE_5SM_ICE_16C_LX330.tcl
"	1xICE + 4x8C *	40	sample_ZSE_5SM_ICE_8C_8C_8C_LX330.tcl
"	1xICE+ 4x16C*	72	sample_ZSE_5SM_ICE_16C_16C_16C_LX330.tcl

* Note that the module which provides the Direct ICE interface is physically the same as the 8C module but configurations for both use models are not compatible.

9.5.2 ZeBu Server 2 Sample Configuration Files

TABLE 28 Available Sample Configuration Files

ZeBu Server 2 System	FPGA Modules	# of FPGAs	Name of Sample Configuration File
2-slot single-unit	1x5F/ICE	5	sample_ZSE2_2S_4C_LX760.tcl
"	2x5F/ICE	10	sample_ZSE2_2S_4C_4C_LX760.tcl

TABLE 28 Available Sample Configuration Files

ZeBu Server 2 System	FPGA Modules	# of FPGAs	Name of Sample Configuration File
"	1x9F/ICE	9	sample_ZSE2_2S_8C_LX760.tcl
"	2x9F/ICE	18	sample_ZSE2_2S_8C_8C_LX760.tcl
"	1x17F	17	sample_ZSE2_2S_16C_LX760.tcl
"	2x17F	34	sample_ZSE2_2S_16C_16C_LX760.tcl
5-slot single-unit	1x17F	17	sample_ZSE2_5S_16C_LX760.tcl
"	2x17F	34	sample_ZSE2_5S_16C_16C_LX760.tcl
"	3x17F	51	sample_ZSE2_5S_16C_16C_16C_LX760.tcl
"	4x17F	68	sample_ZSE2_5S_16C_16C_16C_16C_LX760.tcl
"	5x17F	85	sample_ZSE2_5S_16C_16C_16C_16C_16C_LX760.tcl
"	1x9F/ICE + 1x17F	26	sample_ZSE2_5S_8C_16C_LX760.tcl
"	1x9F/ICE + 4x17F	77	sample_ZSE2_5S_8C_16C_16C_16C_16C_LX760.tcl
"	5x9F/ICE	45	sample_ZSE2_5S_8C_8C_8C_8C_8C_LX760.tcl
5-slot multi unit	1x17F	17	sample_ZSE2_5SM_16C_LX760.tcl
"	2x17F	34	sample_ZSE2_5SM_16C_16C_LX760.tcl
"	3x17F	51	sample_ZSE2_5SM_16C_16C_16C_LX760.tcl

TABLE 28 Available Sample Configuration Files

ZeBu Server 2 System	FPGA Modules	# of FPGAs	Name of Sample Configuration File
"	4x17F	68	sample_ZSE2_5SM_16C_16C_16C_16C_LX760.tcl
"	5x17F	85	sample_ZSE2_5SM_16C_16C_16C_16C_LX760.tcl

9.5.3 ZeBu Server 3 Sample Configuration Files

TABLE 29 Available Sample Configuration Files

ZeBu Server 3 System	FPGA Modules	# of FPGAs	Name of Sample Configuration File
2-slot single-unit	1x9F	9	sample_ZS3_2S_8C_2000T.tcl
"	1x9F/ICE	9	sample_ZS3_2S_ICE_2000T.tcl
"	2x9F/ICE	18	sample_ZS3_2S_ICE_ICE_2000T.tcl
	2x9F	18	sample_ZS3_2S_8C_8C_2000T.tcl
5-slot single-unit	3x9F	27	sample_ZS3_5S_8C_8C_8C_2000T.tcl
"	5x9F	45	sample_ZS3_5S_8C_8C_8C_8C_8C_2000T.tcl
"	5x9F/ICE	45	sample_ZS3_5S_ICE_ICE_ICE_ICE_ICE_ICE_2000T.tcl
5-slot multi unit	5x9F	45	sample_ZS3_5SM_8C_8C_8C_8C_8C_2000T.tcl

9.5.4 ZeBu Server 4 Sample Configuration Files

```
sample_ORION_16U_8H_64x12C_440.tcl  
sample_ORION_2U_1H_8x12C_440.tcl  
sample_ORION_4S_12C_12C_12C_12C_440.tcl  
sample_ORION_4S_12C_440.tcl  
sample_ORION_4S_H6C_440.tcl
```

10 Troubleshooting

This chapter provides information about troubleshooting scenarios and consists of the following sub-sections:

- *Missing Kernel Source Files for the Linux Kernel*
- *ZeBu Server Unit not Detected by The PC*
- *ZeBu Kernel Module Version Mismatch*
- *ZeBu Software Does Not Recognize the Hardware*
- *zInstall Accidentally Called*
- *zServer or zUtils is Running While Trying to Install a Driver*
- *ZeBu Driver Not Installed (Versions prior to L-2016.06)*
- *Problems When Compiling With gcc*
- *Cannot Communicate With Units When a PC is OFF*
- *Runtime Errors Caused by Faulty Memories*
- *Synopsys Installer Does not Install Packages*

10.1 Missing Kernel Source Files for the Linux Kernel

When launching `zInstallLauncher.sh` with automatic compilation of the `zKernel` module you may not have the appropriate source files of the Linux kernel and observe the following message during installation:

```
-- ZeBu : zInstall : Extracting "zKernel" sources from " /home/
<user>/installation_directory/bin" into the local "/zebu/L-
2016.06" directory.
-- ZeBu : zInstall : Compiling "zKernel".
-- ZeBu : zInstall : ERROR : ZINS0101E : Cannot compile sources :
-- ZeBu : zInstall : ERROR : ZINS0100E : make: Entering directory
' /zebu/L-2016.06/zKernel-src '
In directory /zebu/L-2016.06/zKernel-src
Generating config file ... OK (./config-2.6.18-164.el5-x86_64.mk)
Looking for gcc ... OK
Using /usr/bin/gcc: ELF 64-bit LSB executable, AMD x86-64, version
1 (SYSV), for GNU/Linux 2.6.9, dynamically linked (uses shared
libs), for GNU/Linux 2.6.9, stripped
Using built-in specs.
Target: x86_64-redhat-linux
Configured with: ../configure --prefix=/usr --mandir=/usr/share/
man --infodir=/usr/share/info --enable-shared --enable-
threads=posix --enable-checking=release --with-system-zlib --
enable-__cxa_atexit --disable-libunwind-exceptions --enable-
libgcj-multifile --enable-languages=c,c++,objc,obj-
c++,java,fortran,ada --enable-java-awt=gtk --disable-dssi --
enable-plugin --with-java-home=/usr/lib/jvm/java-1.4.2-gcj-
1.4.2.0/jre --with-cpu=generic --host=x86_64-redhat-linux
Thread model: posix
gcc version 4.4.7 Red Hat 6.6)
```

Missing Kernel Source Files for the Linux Kernel

```
Computing sources directory name ... OK
Getting the included files list ... OK
Testing remap_page_range ... FAILED (not mandatory)
Testing remap_pfn_range ... FAILED (not mandatory)
There must be either "remap_page_range" or "remap_pfn_range".
KGCC := /usr/bin/gcc
KERNEL_RELEASE := 2.6.18-164.el5
TGT_SUFFIX := -2.6.18-164.el5-x86_64
KERNEL_FAMILY_2_6 := true
KERNEL_ARCH_DIR := x86_64
make: *** [zKernel.o] Error 255
make: Leaving directory '/zebu/O-2018.09/zKernel-src'
-- ZeBu : zInstall : 16349 close at Mon 6 1 2015 - 14:15:10.
-- ZeBu : zInstall : The zInstall daemon process pid 16349 finished
with exit code 255.
In the install.log file, you can find the following information:
Testing remap_page_range ... FAILED (not mandatory)
Testing remap_pfn_range ... FAILED (not mandatory)
There must be either "remap_page_range" or "remap_pfn_range".
[...]
make: *** [zKernel.o] Error 255
```

In such a case, you must install the Linux package with the source files of your version of the Linux kernel (typical name of package: `kernel-devel-<version>`) and relaunch `zInstallLauncher.sh` to compile the `zKernel` module (see section [Initializing PCIe Boards on Host PC Running ZeBu 2017.03 or Newer Releases](#)).

10.2 ZeBu Server Unit not Detected by The PC

When launching `zInstallLauncher.sh`, the following error message indicates that the PCIe interconnection board of the PC has not been identified:

```
-- ZeBu : zInstall : ERROR : No ZeBu board detected on system!
```

Check that the interconnection board is plugged correctly. If the error persists, send the result of the following command for support (zebu_support@synopsys.com):

```
$ zUtils -getPCInfo
```

10.3 ZeBu Kernel Module Version Mismatch

When loading `zKernel` in the Linux kernel, you may observe this error message:

```
zKernel.yyy.o: kernel-module version mismatch
zKernel.yyy.o was compiled for kernel version yyy while this kernel
is version xxx
```

It is generated when `zKernel` was not compiled for the right Linux kernel version. This happens, for example, when you re-use a module from an installation on another PC that has a different version of the Linux kernel.

Check the kernel version by using `'uname -r'` which reports the kernel version. The version must be equal to the `'yyy'` of the `zKernel.yyy.o` file.

You should check the availability of the Linux kernel source files and relaunch `zInstallLauncher.sh`, as described in section [Initializing PCIe Boards on Host PC Running ZeBu 2017.03 or Newer Releases](#).

Note

When `zKernel` was compiled automatically by `zInstallLauncher.sh`, this error must not occur.

10.4 ZeBu Software Does Not Recognize the Hardware

If you are not using **zSetupSystem** from the latest ZeBu release, the following messages are displayed during the **zSetupSystem** or **zUtils**.

```
-- ZeBu : zUtils : ERROR : LUI2081E : Unit 0 is not detected.  
-- ZeBu : zUtils : ERROR : LUI2338E : Almost 1 PCIe board must be  
connected to an unit.
```

One of the reasons for this issue can be that the software does not recognize the newer version of the hardware.

To resolve this mismatch, you must always use **zSetupSystem** and **zUtils - initSystem** from the latest ZeBu release.

If the error persists, contact Synopsys support.

10.5 zInstall Accidentally Called

10.5.1 Problem Description

The **zSwitchDriver** executable detects manual **zInstall** calls and rejects them with the following message:

```
zSwitchDriver ERROR : zInstall has been used to install release  
<the_release>  
zSwitchDriver ERROR : Removing release <the_release>  
zSwitchDriver ERROR : Please use zInstallLauncher.sh script to  
install ZeBu releases.
```

This message is displayed both on `stderr` and in `zSwitchDriver.0.log`.

10.5.2 Solution

When a **zInstall** is manually run on the workstation, the Device Driver Switching solution errors out. **zInstall** MUST NOT be called manually.

ZeBu releases installed on the workstation must be first cleaned and then installed through the execution of the `zInstallLauncher.sh` as described in section [Installing ZeBu Device Drivers](#).

10.6 zServer or zUtils is Running While Trying to Install a Driver

10.6.1 Problem Description

The following error message is displayed:

```
ERROR : zServer (pid 3165) detected as running.  
ERROR : /my_path/my_release INSTALLATION ABORTED.
```

It means that a **zUtils** or **zServer** process is running while calling the following command:

```
zInstallLauncher.sh -start -r <zebu_release_path>
```

10.6.2 Solution

Wait for the end of **zUtils** and/or **zServer** execution and retry the installation command.

10.7 ZeBu Driver Not Installed (Versions prior to L-2016.06)

10.7.1 Problem Description

If the driver for a ZeBu release is not installed when starting a run, you receive the following error message:

```
-- ZeBu : zServer : ERROR : LUI0962E : The daemon for your release  
is not installed, please restart "zInstall".  
  
-- ZeBu : theTest : ERROR : The connection is closed, signal 15  
(TERM : "Terminated") has been trapped..
```

10.7.2 Solution

Install the required driver for the release with the `zInstallLauncher.sh` script as described in section [Installing ZeBu Device Drivers](#).

Note

Do not use `zInstall` as prompted by the error message above. `zInstall` MUST NOT be called manually.

10.8 Problems When Compiling With gcc

The gcc compiler, version 4.4 is required for correct installation of the ZeBu software. If you cannot run gcc, you should first verify that a gcc is installed on the PC (usually, gcc can be found in /usr/local/bin or in /usr/bin):

```
$ ls /usr/bin/gcc* /usr/local/bin/gcc*
```

If no gcc can be found in the above locations, your Linux installation is not a developer installation. Install the gcc from your Linux distribution.

For example, using Red Hat Linux, this operation can be done using `gnorpm` or the `rpm` command. The required packages for gcc can be found in a section called Development/Language.

When gcc is installed, you must verify that your environment recognizes it. For that, you can verify the version of gcc:

```
$ gcc -v
gcc version 4.4.7 Red Hat 6.6)
```

If no gcc is found, verify that the `$PATH` environment variable contains the directory where gcc is installed, and if it does not contain it, add it as follows:

```
$ gcc -v
bash: gcc: command not found
$ echo $PATH
/usr/bin:/bin:/sbin:/usr/local/tools/simulators
```

10.9 Cannot Communicate With Units When a PC is OFF

In a multi user system configuration (single or multi unit), if one of the host PCs connected to any unit is switched OFF, you may no longer can communicate with the units from any PC connected to the system. In such a case, the following error is displayed:

```
-- ZeBu : zUtils : Loading U0_BP_FC0 with          "/home/<user>/
installation_directory/etc/firmwares/ZSE/default/
zse_sbp_2s_fc.bit" (using CPU).

..... LAU0331E : Cannot check if U0_BP_FC0 is
done (cannot read status).

-- ZeBu : zUtils : ERROR : FAILED.

-- ZeBu : zUtils : WARNING : Retrying to load U0_BP_FC0 (1/10).
-- ZeBu : zUtils : Loading U0_BP_FC0 with "/home/<user>/
installation_directory/etc/firmwares/ZSE/default/
zse_sbp_2s_fc.bit" (using CPU).

LAU0330E : Cannot reset U0_BP_FC0 (cannot check cmd 0x00000002).
-- ZeBu : zUtils : ERROR : FAILED.
```

In most cases, communication resumes as soon as ALL the PCs are switched ON. If not, you should power OFF/ON all units and launch **zSetupSystem** (described in section [zSetupSystem Tool](#)) or **zUtils -initSystem** (described in section [Initializing the System With zUtils -initSystem](#)).

10.10 Runtime Errors Caused by Faulty Memories

A design using a memory declared as faulty during setup (see [Testing the Memories](#)) causes the runtime software to error out with one of the following messages, based on the memory type.

10.10.1 Faulty DDR2 Memory for the SRAM Trace Memory

```
LUI1971E: Trace Memory U0_M0_FS_DDR2_0 has been detected as  
"faulty" during setup and the design is trying to use it
```

10.10.2 Faulty DDR2 and RLDRAM Memories

```
LUI1969E: Memory U0_M0_F04_DDR2_0 has been detected as "faulty"  
during setup and the design is trying to use it  
LUI1970E: Memory U0_M0_F00_RLDRAM_0 has been detected as "faulty"  
during setup and the design is trying to use it
```

10.10.3 Faulty Mx_FS Memory

If the Mx_FS memory is found to be faulty when the design is being loaded:

```
Warning : Memory U0_M0_FS has been detected as "faulty":  
Warning : No bitstream cache will be available for this module  
Warning : No SRAM_TRACE will be available for this module
```

Runtime Errors Caused by Faulty Memories

```
# 5- TRACE_SIZE must be modulo 1024; else, it will be rounded down  
  
Warning : Specified TRACE_SIZE (0x.....) is not modulo 1024  
Warning : TRACE_SIZE is rounded down to 0x.....
```

If the specified trace size is less than 1024:

```
Warning : Specified TRACE_SIZE (0x....) is too small (MIN = 0x400)  
Warning : TRACE is disabled
```

If the trace size exceeds that of the module's physical memory:

```
Warning : Specified TRACE_SIZE (0x....) exceeded MAX TRACE_SIZE for  
Ux_Mx module types  
  
Warning : TRACE Size is forced to 0x.... : No bitstream cache will  
be available for this Module
```

10.10.4 Faulty DDR3 Memories

```
LUI1969E : Memory Unit 0 Module 0 (physical Unit 0 Module 1)  
(zse_xc7v_8c_2000_v1) F00 DDR 0 has been detected as "faulty"  
during setup and the design trying to use it
```

10.10.5 Faulty Mx_FS DDR3 Memory

If the Mx_FS DDR3 memory is found to be faulty when the design is being loaded:

- Mx_FS DDR3 memory for SRAM trace (static-probes)

```
WARNING : U0_M0_FS_DDR_0 memory has been detected as "faulty" :  
WARNING : - No SRAM_TRACE will be available for this module
```

- Mx_FS DDR3 memory for bitstream cache

```
WARNING : U0_M0_FS_DDR_1 memory has been detected as "faulty" :  
WARNING : - No Cache bitstream will be available for this module
```

10.11 Synopsys Installer Does not Install Packages

10.11.1 Problem Description

Synopsys installer does not install the packages if the following warning appears:

```
Installer: Warning Executing zebu_xilinx_vivado_2018_1_patched_vO-2018.09_common.spf.csh failed, EST file zebu_xilinx_vivado_2017_3_patched_vO-2018.09_common.spf is not created
Installer: Creating EST file zebu_vO-2018.09_common.spf ...
Installer: Warning Executing zebu_vO-2018.09_common.spf.csh failed, EST file zebu_vO-2018.09_common.spf is not created
Installer: Creating EST file zebu_xilinx_vivado_2018_1_vO-2018.09_common.spf ...
Installer: Warning Executing zebu_xilinx_vivado_2018_1_vO-2018.09_common.spf.csh failed, EST file zebu_xilinx_vivado_2018_1_vO-2018.09_common.spf is not created
Installer: Following EST file(s) have been ignored due to previous errors or invalid characters in filename
Installer:      zebu_xilinx_vivado_2018_1_patched_vO-2018.09_common.spf
Installer:      zebu_vO-2018.09_common.spf
Installer:      zebu_xilinx_vivado_2018_1_vO-2018.09_common.spf
```

This error occurs because the .csh files do not exist or Unix execution rights has been set in the .csh file.

10.11.2 Solution

- Check if the `<package_name>.csh` file exists.
- Ensure that the `.csh` file has execution rights permissions.

11 Appendix A: Technical Data

This chapter provides the technical data about the ZeBu Server hardware, and consists of the following sections:

- *Size and Weight*
- *Environment*
- *Power Supply*
- *PCIe Interconnection Board and Cable*
- *Power Cords*

11.1 Size and Weight

11.1.1 ZeBu Server 1, ZeBu Server 2, and ZeBu Server 3

TABLE 30 Size and Weight for ZeBu Server 1, ZeBu Server 2, and ZeBu Server 3

	Weight		W x D x H (*)	
ZeBu Server 2-slot unit	< 30 kg	< 65 lbs	50cm x 50cm x 22cm	19.7" x 19.7" x 8.7"
ZeBu Server 5-slot unit	< 70 kg	< 155 lbs	50cm x 50cm x 51cm	19.7" x 19.7" x 20.1"
ZeBu Server Hub	< 10 kg	< 22 lbs	40cm x 40cm x 8.5cm	15.7" x 15.7" x 3.3"
PCIe interconnection board			16.8cm x 11.2cm	6.6" x 4.4"
(*) including handles & feet				

Note For 5-slot units, the shelf must be carefully chosen to support the weight of the system.

11.1.2 ZeBu Server 4

For details on size and weight of ZeBu Server 4, see the *ZeBu Server 4 Site Planning Guide*.

11.1.3 Racks and Cabinets

11.1.3.1 ZeBu Server 1, ZeBu Server 2, and ZeBu Server 3

Synopsys recommends installing ZeBu Server units and host PCs in separate racks. To ensure proper airflow from right to left of your ZeBu Server, follow the cabinet requirements:

TABLE 31 Racks and Cabinet Recommendations

Cabinet Width	72 cm
Lateral Clearance	10 cm (4") on the right side for airflow
Front Clearance (for maintenance)	1 m (39.3")
Rear Clearance (for maintenance)	1m (39.3") - depends on lab organization and number of interconnection cables.
Right Panel	Right panel must be perforated
Left Panel	Left panel must be solid
Airflow	Three fans with a 535 CFM airflow (1605 CFM total) must be installed on the top-left corner of the cabinet
Number of ZeBu Server Units per Cabinet	Up to three ZeBu Server units

Note

For a multi unit configuration, contact your Synopsys representative.

11.1.3.2 ZeBu Server 4

For details on racks and cabinets of ZeBu Server 4, see the *ZeBu Server 4 Site Planning Guide*.

11.2 Environment

11.2.1 Uninterruptible Power Supply

The whole server room must be equipped with an uninterruptible power supply (UPS).

11.2.2 Environmental Constraints

The operating temperature is a critical point for ZeBu Server. For a proper air circulation on the ZeBu Server, Synopsys recommends installing it in an air-conditioned room with sufficient clearances on both left and right sides.

TABLE 32 Environmental Constraints

Operating ambient temperature	10°C to 20°C (50°F to 68°F)
Operating relative humidity	Up to 80% non-condensing
Cooling for a 2-slot unit	2,720 BTUs (for total power of 1 kW)
Cooling for a 5-slot unit	6,800 BTUs (for total power of 2 kW)
Cooling for a 2-unit system	14,200 BTUs (for total power of 4 kW)
Cooling for a 3-unit system	21,000 BTUs (for total power of 6 kW)
Cooling for a 4-unit system	27,800 BTUs (for total power of 8 kW)
Cooling for a 5-unit system	34,600 BTUs (for total power of 10 kW)
Noise generated by a 2-slot unit	ZeBu Server 1: 60 dBA ZeBu Server 2: 85 dBA ZeBu Server 3: 85 dBA
Noise generated by a 5-slot unit	ZeBu Server 1: 70 dBA ZeBu Server 2: 80 dBA ZeBu Server 3: 80 dBA
Operating altitude	Up to 2,000 m (up to 6,500 feet)
Installation	Category II

Environment

TABLE 32 Environmental Constraints

Pollution	Pollution degree 2
RoHS	Compliant with Directive 2002/95/EC

11.2.3 Automatic Temperature Control

If the temperature of an FPGA on which the design is run exceeds 85°C (185°F), the emulation stops and the temperatures of all the FPGAs belonging to the module are displayed in Celsius degrees:

```
Error : ===== Overheat detected (0x00000004) =====
Error : --- U0_M0 ---
Error : Type      : zse_xc7v_8c_2000_v1
Error : ID       : 0x0092
Error : M0 (physical M0) FC : 71 C
Error : M0 (physical M0) FS : 63 C
Error : M0 (physical M0) IF : 93 C
Error : M0 (physical M0) F00 : 72 C
Error : M0 (physical M0) F01 : 71 C
Error : M0 (physical M0) F02 : 75 C
Error : M0 (physical M0) F03 : 45 C
Error : M0 (physical M0) F04 : 43 C
Error : M0 (physical M0) F05 : 62 C
Error : M0 (physical M0) F06 : 68 C
Error : M0 (physical M0) F07 : 43 C
Error : M0 (physical M0) F08 : 45 C
```

where:

- 0x00000010 is an internal register.

- 0x0060 is the ID of the overheated module.

In the example above, users cannot connect to module M0 if the temperature of IF FPGA has not dropped below 85 °C (185 °F).

Note

A temperature log file, `temperature_<yyy_mm_dd>.log`, is described in detail in section [connections_YYYY_MM_DD.xml](#).

11.3 Power Supply

11.3.1 For ZeBu Server 1

The current limitations of the system due to the power supply fuse are:

- 10 Amp for a 2-slot ZeBu Server 1 unit
- 12.5 Amp for a 5-slot ZeBu Server 1 unit

ZeBu Server 1 2-slot Unit	100–240 V, 50-60 Hz, single phase
ZeBu Server 1 5-slot Unit	200–240 V, 50-60 Hz, single phase

TABLE 33 Power Consumption for a System With up to 32 FPGAs

Mean Power Consumption	400 W
Maximum Power Consumption*	1,500 W

TABLE 34 Power Consumption for a System With More Than 32 FPGAs

Mean Power Consumption	700 W
Maximum Power Consumption*	3,000 W
* Estimated Instantaneous Power	

Power Supply

In any case, the power consumption of your ZeBu Server 1 unit depends on:

- The number of FPGAs in the unit
- The number of used FPGAs for the DUT
- The fill rate and toggle rate of your DUT FPGAs
- The initial state of the design, the memory requirements
- The system clock and design clocks frequencies

For a given application, the mean power measurement can give variable results, always lower than the values given in the preceding tables, but the mains supply must be chosen based on the maximum consumption.

11.3.2 For ZeBu Server 2

The current limitations of the system due to the power supply fuse are:

- 10 Amp for a 2-slot ZeBu Server 2 unit
- 15 Amp for a 5-slot ZeBu Server 2 unit

TABLE 35 Voltage Range for the ZeBu Server 2 Unit

ZeBu Server 2 2-slot Unit	100–240 V, 50-60 Hz, single phase
ZeBu Server 2 5-slot Unit	200–240 V, 50-60 Hz, single phase

TABLE 36 Maximum Power Consumption

Empty 2-slot Chassis	100 W
Empty 5-slot Chassis	300 W
5F/ICE Module	200 W
9F/ICE Module	400 W
17F Module	500 W

Example:

For a single-unit ZeBu Server 2 system with a 5-slot chassis and 5x17F modules, the maximum power consumption would be:

$$300 + 5 \times 500 = 2800 \text{ W}$$

In any case, the power consumption of your ZeBu Server 2 unit depends on:

- The number of FPGAs in the unit
- The number of used FPGAs for the DUT
- The fill rate and toggle rate of your DUT FPGAs
- The initial state of the design, the memory requirements
- The system clock and design clocks frequencies

For a given application, the mean power measurement can give variable results, always lower than the values given in the preceding tables, but the mains supply must be chosen based on the maximum consumption.

11.3.3 For ZeBu Server 3

The current limitations of the system due to the power supply fuse are:

- 10 Amp for a 2-slot ZeBu Server 3 unit
- 15 Amp for a 5-slot ZeBu Server 3 unit

TABLE 37 Voltage Range for the ZeBu Server 3 Unit

ZeBu Server 3 2-slot Unit	100–240 V, 50-60 Hz, single phase
ZeBu Server 3 5-slot Unit	200–240 V, 50-60 Hz, single phase

TABLE 38 Maximum Power Consumption

Empty 2-slot Chassis	100 W
Empty 5-slot Chassis	300 W
9F Module	350 W
9F/ICE Module	400 W

Example:

For a single-unit ZeBu Server 3 system with a 5-slot chassis and 5x9F modules, the maximum power consumption would be:

$$300 + 5 \times 350 = 2050 \text{ W}$$

In any case, the power consumption of your ZeBu Server 3 unit depends on:

- The number of FPGAs in the unit
- The number of used FPGAs for the DUT
- The fill rate and toggle rate of your DUT FPGAs
- The initial state of the design, the memory requirements
- The system clock and design clocks frequencies

For a given application, the mean power measurement can give variable results, always lower than the values given in the preceding tables, but the mains supply must be chosen based on the maximum consumption.

11.3.4 For ZeBu Server 4

For details on ZeBu Server 4 power supply, see the *ZeBu Server 4 Site Planning Guide*.

11.4 PCIe Interconnection Board and Cable

11.4.1 ZeBu Server 1, ZeBu Server 2, and ZeBu Server 3

The PCIe interconnection board for ZeBu Server is compliant with PCI Express 2.0 standard.

TABLE 39 Dimensions of the PCIe Interconnection Board

	L x W	
PCIe interconnection board	16.8cm x 11.2cm	6.6" x 4.4"

The PCIe board has eight physical lanes but only 4 are used. It can fit into:

- Any 8- or 16-lane PCIe slot
- Some 4-lane PCIe slots

The PCIe cable length is three meters (9.8 feet) (optionally 5 meters).

11.4.2 ZeBu Server 4

The PCIe interconnection board for ZeBu Server 4 is compliant with PCI Express 3.0 standard.

TABLE 40 Dimensions of the PCIe Interconnection Board

	L x W	
PCIe interconnection board	16.8cm x 11.2cm	6.6" x 4.4"

The PCIe board has eight lanes. It can fit into any 8- or 16-lane PCIe slot.
The PCIe cable length is three meters (9.8 feet).

Note

The BIOS must be modified to set the fans to be always on.

11.5 Power Cords

11.5.1 ZeBu Server 1, ZeBu Server 2, and ZeBu Server 3

Depending on the region and the chassis, each ZeBu Server unit is shipped with the appropriate power cord. Synopsys recommends equipping your lab with power sockets corresponding to the shipped cable.

The power cord length varies between 2.5 meters (8 feet) and 3.5 meters (11.5 feet).

For information regarding the power cord, considering your region and your ZeBu Server unit, see [Table 41](#). For any additional question, contact your Synopsys representative.

TABLE 41 ZeBu Server Power Cables

Region	Power Cord for a ZeBu Server 2-slot Unit (Male to Female)	Power Cord for a ZeBu Server 5-slot Unit (Male to Female)
North America	NEMA 5-15P to IEC-60320-C19	LOCKING NEMA L6-30P to IEC-60320-C19
Europe (except UK)	CEE 7/7 EUROPEAN SCHUKO to IEC-60320-C19	IEC60309-32A-250V To IEC-60320-C19
Japan	JIS 8303 JAPAN To IEC-60320-C19	LOCKING NEMA L6-30P JAPAN To IEC-60320-C19
Rest of the world	NEMA 5-15P to IEC-60320-C19 Or CEE 7/7 EUROPEAN SCHUKO to IEC-60320-C19	LOCKING NEMA L6-30P to IEC-60320-C19 Or IEC60309-32A-250V To IEC-60320-C19

11.5.2 ZeBu Server 4

For details on ZeBu Server 4 power cords, see the *ZeBu Server 4 Site Planning Guide*.

12 Appendix B: ZeBu Device Driver Switching Solution

This appendix covers the following topics:

- [Overview](#)
- [Package Description](#)
- [Device Driver Switching Solution](#)
- [Troubleshooting](#)

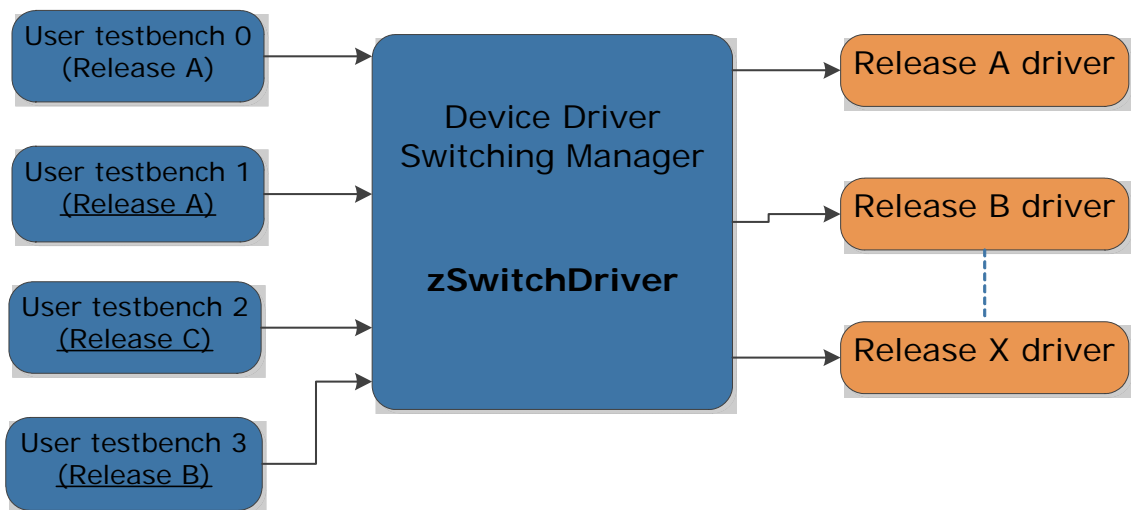
12.1 Overview

This ZeBu Device Driver Switching Solution:

- Allows installation of drivers for multiple ZeBu releases by sharing the same ZeBu hardware system,
- Centralizes the Device Driver Switching mechanism in a unique process, and
- Automatically compiles the ZeBu loadable kernel module (**zKernel**,) if necessary.

The following block diagram explains ZeBu Device Driver Switching solution.

FIGURE 10. Device Driver Switching Description



12.1.1 Package Description

The following different components are provided within the package:

- | -- README.txt
- | -- zInstallLauncher.sh
- `-- zSwitchDriver

12.1.1.1 zSwitchDriver

zSwitchDriver is the executable that replaces all **zInstall** daemons and manages the ZeBu release switching.

The **zSwitchDriver** executable MUST NOT be launched manually: it is automatically handled by the provided `zInstallLauncher.sh` script.

12.1.1.2 zInstallLauncher.sh

The Device Driver Switching solution is based on a script called `zInstallLauncher.sh`. This script must be used instead of **zInstall** to manage the ZeBu Device Driver installation on your workstation.

For more information on its use model and scenarios, see chapter [Device Driver Switching Solution](#).

12.1.2 Device Driver Switching Solution

The `zInstallLauncher.sh` script of the Device Driver Switching solution features the following available operations:

- Cleaning any previous ZeBu environment, see section [Cleaning Previous Releases From Your ZeBu Environment](#).
- Installing a new device driver, see section [Installing a New Device Driver](#).
- Uninstalling a device driver, see section [Removing a Device Driver](#).
- Displaying the list of installed releases, see section [Displaying the List of Installed Releases](#).

12.1.2.1 Cleaning Previous Releases From Your ZeBu Environment

Cleaning previous releases from your Zebu environment prepares the workstation before a new installation of a ZeBu release. This operation is required only before any new installation.

To clean your ZeBu environment, launch the following command with root permissions:

```
zInstallLauncher.sh -clean
```

12.1.2.2 Installing a New Device Driver

To install a new device driver for a release, launch the following command with root permissions:

```
zInstallLauncher.sh -start -r <zebu_release_path>  
[-driverPath <driver_package_path>/drivers/zebu_zs]
```

where:

- `zebu_release_path` corresponds to the `$ZEBU_ROOT` of the ZeBu release.
- `-driverPath` is optional. This option is only available if the release is 2017.03 or later. It is required to indicate the path of the `zebu_zs` driver, if the package hierarchy is modified (directories shuffled or moved from the Zebu drivers package).

Example

```
zInstallLauncher.sh -start -r /rel_path/the_release
```

12.1.2.3 Removing a Device Driver

To remove the device driver from a ZeBu release, launch the following command with root permissions:

```
zInstallLauncher.sh -stop -r <zebu_release_path>
```

where:

- `zebu_release_path` corresponds to the `$ZEBU_ROOT` of the ZeBu release.

12.1.2.4 Displaying the List of Installed Releases

To display the list of installed ZeBu releases, launch the following command:

```
zInstallLauncher.sh -list
```


Example


```
Num releases installed: 1  
/depot/release_installation/zebu_2015.09/amd64
```


12.1.2.5 Use Case Scenarios

Installing ZeBu Device Drivers

Note

 All ZeBu emulation runs must be stopped during this installation process.

 All commands below must be launched with root permissions.

To install the ZeBu Device Drivers, perform the following steps:

6. Clean your workstation using the following command:

```
zInstallLauncher.sh -clean
```

NOTE: This execution is required only when preparing the PC for ZeBu for the first time.

7. Install your ZeBu Device Drivers with the `-start` option as provided in the following examples:

```
zInstallLauncher.sh -start -r <zebu_release_1_path>
```

```
zInstallLauncher.sh -start -r <zebu_release_2_path>
```

```
zInstallLauncher.sh -start -r <zebu_release_N_path>
```

where:

- ❑ `zebu_release_n_path` corresponds to the `$ZEBU_ROOT` of the ZeBu releases.

The system is ready for use.

Removing ZeBu Releases

You must stop all ZeBu emulation runs during this installation process.

- Launch the following command with root permissions to remove any release as follows:

```
zInstallLauncher.sh -stop -r <zebu_release_path>
```

where:

- `zebu_release_path` corresponds to the `$ZEBU_ROOT` of the ZeBu release.

Viewing Release Switching Logs

The **zSwitchDriver** process stores information such as the history of the release switching and other error messages in a file named `/zebu/zSwitchDriver_messages.<N>.log`.

where N goes from 0 to 10:

- 0 corresponds to the current running **zSwitchDriver** process
- Older runs are stored from 1 to 10

Note

The zSwitchDriver process stores information only for ZeBu Server 1, 2, and 3.

12.1.3 Troubleshooting

For troubleshooting information on the ZeBu Device Driver Switching solution, see the following sections:

- [ZeBu Server Unit not Detected by The PC](#)
- [ZeBu Kernel Module Version Mismatch](#)
- [ZeBu Driver Not Installed \(Versions prior to L-2016.06\)](#)

13 Appendix C:

setup_template.zini

This chapter provides information about the `setup_template.zini` file for:

- [ZeBu Server 1](#)
- [ZeBu Server 2](#)
- [ZeBu Server 3](#)

13.1 ZeBu Server 1

```
# This file will be parsed with one of the following tools and
options:
# - zSetupSystem <my_setup>.zini
# - zUtils -initSystem <my_setup>.zini

#####
##### MANDATORY DECLARATIONS #####
#####

##### ZEBU_SYSTEM_DIR: output directory for the system files
generated by zUtils #####
# All these system files will be used by the ZeBu Server system at
runtime.
# The following directories will be generated (or updated if they
already exist):
#   - dt          : contains all input files for 'zUtils -
calibration' and 'zUtils -dt'
#                   user tests
#   - calibration : contains all calibration files
#   - history      : contains FLLP (Frequency-Limited LVDS Pairs) or
faulty pins history files
#   - config       : output directory generated by zConfig
# The following files will be generated (or updated if they already
exist):
#   - config.dff   : complete architecture of the detected ZeBu
Server system
```

```
# "patchPath": input path of the diagnostics directory bitstreams
# This file/directory is located right where you installed your
system's configuration patch
# A copy of patchFile will be made in $ZEBU_SYSTEM_DIR/dt/V5.0 (see
above)
# A link of the patchPath/bitstreams will be made in
$ZEBU_SYSTEM_DIR/dt/V5.0 (see above)
# Use $copyBitstreams to request a copy instead of a link (see
below)

# Example:
#$patchFile = "/zebu/release/etc/firmwares/ZSE/dt/V5.0/
patch_file.diag";
#$patchPath = "/zebu/release/etc/firmwares/ZSE/dt/V5.0/";

# NOTE 1: if $patchFile is not specified:
# zUtils will look for patchFile in the $ZEBU_ROOT/etc/firmwares/
ZSE/dt/V5.0 directory.
# This will work with ONLY ONE patchFile present in this directory
(i.e. ONLY ONE diagnostics
# patch was installed)
# NOTE 2: if $patchPath is not specified:
# zUtils will look for diagnostics directory bitstreams in the
# $ZEBU_ROOT/etc/firmwares/ZSE/dt/V5.0 directory.
```

```
##### Input system cables configuration file #####
# This input file will only be taken into account with a 5Sx multi-
unit system configuration.
# "cables_xU.dff" is the input file which defines inter-unit system
cables connections.
# 'xU' is the number of units in the ZeBu Server system.
# This file is present in the release $ZEBU_ROOT/etc/
configurations.
# The generated config.dff file (see above) will contain a copy of
this file.
# Example for a '2-unit' ZeBu Server system configuration
#$systemCableFile = "/zebu/release/etc/configurations/
cables_2U.dff";
# NOTE: if $systemCableFile is not specified:
# zUtils will look for the input system cables configuration file
in $ZEBU_ROOT/etc/configurations/cables_xU.dff which is compatible
with patchFile
#####
##### TRACE DECLARATIONS (Static Probes)#####
#####

# Maximum TRACE_SIZE according to module type :
#   - 16C : 4 GBytes = 0x40000000 (32-bit words)
#   - 8C : 2 GBytes = 0x20000000 (32-bit words)
#   - 4C : 4 GBytes = 0x40000000 (32-bit words)
# 1- TRACE_SIZE must be defined in 32-bit words
# 2- TRACE_SIZE is defined for the complete system : All modules
use this size
```

```

# 3- If the TRACE_SIZE exceeds the module's maximum size, then the
entire memory of the
#   module will be used for TRACE (no Cache BitStream)
# 4- It is possible to set TRACE_SIZE to 0x0, in this case :
#   a- The complete memory will be used for Cache BitStream
#   b- TRACE feature will be disabled
# 5- TRACE_SIZE must be modulo 1024, else, it will be rounded down
# 6- If TRACE_SIZE is modified, it will only be taking into account
after the next

#   "zUtils -initSystem"
# Default TRACE_SIZE : 256 MBytes = 0x4000000 (32-bit words)
$TRACE_SIZE = 0x4000000;
#####
##### MISC. DECLARATIONS #####
#####

# Do not launch memories tests at the end of setup:
#$noMemTest = 1;

# Copy the diagnostics bitstreams directory instead of the link:
#$copyBitstreams = 1;

# Do not launch zConfig at the end of setup:
#$zConfig = 0;

```

```
# Do not use $ZEBU_SYSTEM_DIR/history (faulty pins history)
#$noHistory = 1;

# Specify default Unit #num Smarti Z-ICE Vcc (1.5 V by default)
# See below for available power supply voltages (e.g. Unit 0)
# Select one of the following lines:
#$U0.SmartZice.Vcc = "1.5 V";
#$U0.SmartZice.Vcc = "1.8 V";
#$U0.SmartZice.Vcc = "2.5 V";
#$U0.SmartZice.Vcc = "3.3 V";

# Specify default Unit #num Module #num 8C-ICE Vcc Ios #num (1.5 V
by default)
# See below for available power supply voltages (e.g. Unit 0 Module
0 VccIo 1 and 2)
# Select one lines for each Vcc per 8C-ICE Module:
#$U0.M0.DirectIce.VccIo_1 = "1.5 V";
#$U0.M0.Directice.VccIo_1 = "1.8 V";
#$U0.M0.Directice.VccIo_1 = "2.5 V";
#$U0.M0.Directice.VccIo_1 = "3.3 V";
#$U0.M0.DirectIce.VccIo_2 = "1.5 V";
#$U0.M0.Directice.VccIo_2 = "1.8 V";
#$U0.M0.Directice.VccIo_2 = "2.5 V";
#$U0.M0.Directice.VccIo_2 = "3.3 V";
```


13.2 ZeBu Server 2

```
# This file is a template to help creating your own <my_setup>.zini
file

# <my_setup>.zini file is an input for the following tools:
# - zSetupSystem <my_setup>.zini
# - zUtils -initSystem <my_setup>.zini

#####
##### MANDATORY DECLARATIONS #####
#####

##### ZEBU_SYSTEM_DIR: name of the setup directory (output
directory for zSetupSystem #####
# All these system files are used by the ZeBu system at runtime.
# The following directories are generated (or updated if they
already exist):
#   - dt           : contains all input files for 'zUtils -
calibration' and 'zUtils -dt' user tests
#   - calibration : contains all calibration files
#   - history      : contains FLLP (Frequency-Limited LVDS Pairs)
history files
#   - config       : output directory generated by zConfig
# The following files are generated (or updated if they already
exist):
#   - config.dff    : complete architecture of the detected ZeBu
system
#   - memories.dff  : status of all the memories in the ZeBu system
#   - status_se.tcl : system status for the design compilation
process (input file for zConfig)
```

```
# - <my_setup>.zini      : a copy of the setup file with the same
name as the file used for zSetupSystem

$ZEBU_SYSTEM_DIR = "my_ZeBu_system";

#####

##### OPTIONAL DECLARATIONS #####

#####

##### Input configuration diagnostics file #####
# "patchFile": diagnostics file which is applicable for the
ZeBu system configuration
# "patchPath": path to the directory in which the diagnostics
files are stored
# This file and directory are located right where you
installed the diagnostics patch for your system
# A copy of patchFile is made in $ZEBU_SYSTEM_DIR/dt/
<diag_version>
# A link of the patchPath/bitstreams is created in
$ZEBU_SYSTEM_DIR/dt/<diag_version>
# Use $copyBitstreams to request a copy instead of a link

# NOTE 1: if $patchFile is not specified:
#           zUtils checks in $patchPath directory and use the
diagnostics file found in this directory if there is ONLY ONE
diagnostics file.
#           If several diagnostics files are present in the
$patchPath directory, zUtils errors out.
# NOTE 2: if $patchPath is not specified:
```

```
#          zUtils looks for diagnostics files in the $ZEBU_ROOT/etc/
firmwares/ZSE/dt/V5.6 directory.

# Example:
#$patchFile = "/zebu/release/etc/firmwares/ZSE/dt/V5.6/
patch_file.diag";
#$patchPath = "/zebu/release/etc/firmwares/ZSE/dt/V5.6/";
#$copyBitstreams = 1;

##### Input system cables configuration file #####
# For a multi-unit system, the 'cables_xU.dff' file is necessary to
describe the interconnection between units.
# 'xU' is the number of units in the ZeBu Server system.
# This file is available in the release $ZEBU_ROOT/etc/
configurations.
# The config.dff file generated by zSetupSystem contains a copy of
this file.
# NOTE: $systemCableFile s optional: the appropriate cable_xU.dff
for your system is searched by default in the $ZEBU_ROOT/etc/
configurations/ directory

# Example for a '2-unit' system in a different directory
#$systemCableFile = "/zebu/release/etc/configurations/
cables_2U.dff";

#####
##### TRACE DECLARATIONS (Static Probes) #####
#####
```

```

# Maximum TRACE_SIZE according to module type :
#   - 17F module      : 4 GBytes = 0x40000000 (32-bit words)
#   - 9F/ICE module   : 4 GBytes = 0x40000000 (32-bit words)
#   - 5F/ICE module   : 4 GBytes = 0x40000000 (32-bit words)

# 1- TRACE_SIZE must be defined in 32-bit words
# 2- TRACE_SIZE is defined for the complete system : All modules
use this size
# 3- If the TRACE_SIZE exceeds the module's maximum size, then the
entire memory of the module is used for TRACE (no Cache BitStream)
# 4- It is possible to set TRACE_SIZE to 0x0, in this case :
#   a- The complete memory is used for Cache BitStream
#   b- TRACE feature is disabled
# 5- TRACE_SIZE must be modulo 1024, else, it is rounded down
# 6- If TRACE_SIZE is modified, it is taken into account after the
next "zUtils -initSystem"

# Default TRACE_SIZE : 256 MBytes = 0x4000000 (32-bit words)
$TRACE_SIZE = 0x4000000;
#####
##### MISC. DECLARATIONS #####
#####

# Do not launch memories tests at the end of setup:
#$noMemTest = 1;

# Do not launch zConfig at the end of setup:
#$zConfig = 0;

```

```
# Do not use $ZEBU_SYSTEM_DIR/history for FLLP (Frequency-Limited
LVDS Pairs)
#$noHistory = 1;

# Specify the Smart Z-ICE Vcc voltage for a given unit (1.5 V by
default)
    # See below for available power supply voltages (e.g. Unit 0)
    # Select one of the following lines:
    #$U0.SmartZice.Vcc = "1.5 V";
    #$U0.SmartZice.Vcc = "1.8 V";
    #$U0.SmartZice.Vcc = "2.5 V";
    #$U0.SmartZice.Vcc = "3.3 V";

# Specify the Direct ICE VccIo voltage for a given unit and
module
# 1.2 V by default on ZeBu Server 2 and ZeBu Blade 2
# See below for available power supply voltages for Unit0
Module0
# Uncomment the appropriate line to change VccIo_1 and/or
VccIo_2:
#$U0.M0.DirectIce.VccIo_1 = "1.2 V";
#$U0.M0.DirectIce.VccIo_1 = "1.5 V";
#$U0.M0.Directice.VccIo_1 = "1.8 V";
#$U0.M0.Directice.VccIo_1 = "2.5 V";
#$U0.M0.DirectIce.VccIo_2 = "1.2 V";
#$U0.M0.DirectIce.VccIo_2 = "1.5 V";
#$U0.M0.Directice.VccIo_2 = "1.8 V";
#$U0.M0.Directice.VccIo_2 = "2.5 V";
```

13.3 ZeBu Server 3

```
# This file will be parsed with one of the following tools and
options:
# - zSetupSystem <my_setup>.zini
# - zUtils -initSystem <my_setup>.zini

#####
##### MANDATORY DECLARATIONS #####
#####

##### ZEBU_SYSTEM_DIR: output directory for the system files
generated by zUtils #####
# All these system files will be used by the ZeBu Server system at
runtime.
# The following directories will be generated (or updated if they
already exist):
#   - dt           : contains all input files for 'zUtils -
calibration' and 'zUtils -dt' user tests
#   - calibration : contains all calibration files
#   - history      : contains FLLP (Frequency-Limited LVDS Pairs) or
faulty pins history files
#   - config       : output directory generated by zConfig
# The following files will be generated (or updated if they already
exist):
#   - config.dff    : complete architecture of the detected ZeBu
Server system
#   - memories.dff  : status of each memory in the ZeBu Server
system
```

```

# - status_se.tcl : system status for the design compilation
process (input file for zConfig)

# - setup.zini      : a copy of my_setup.zini
$ZEBU_SYSTEM_DIR = "my_ZeBu_system";

#####
##### OPTIONAL DECLARATIONS #####
#####

##### Input configuration diagnostics file #####
# "patchFile": input file which defines the ZeBu Server system
configuration
# "patchPath": input path of the diagnostics directory bitstreams
# This file/directory is located right where you installed your
system's configuration patch
# A copy of patchFile will be made in $ZEBU_SYSTEM_DIR/dt/V6.0 (see
above)
# A link of the patchPath/bitstreams will be made in
$ZEBU_SYSTEM_DIR/dt/V6.0 (see above)
# Use $copyBitstreams to request a copy instead of a link (see
below)

# Example:
#$patchFile = "/zebu/release/etc/firmwares/ZSE/dt/V6.0/
patch_file.diag";
#$patchPath = "/zebu/release/etc/firmwares/ZSE/dt/V6.0/";
# NOTE 1: if $patchFile is not specified:
# zUtils will look for patchFile in the $ZEBU_ROOT/etc/firmwares/
ZSE/dt/V6.0 directory.
# This will work with ONLY ONE patchFile present in this directory
(i.e. ONLY ONE diagnostics patch was installed)

```

```

# NOTE 2: if $patchPath is not specified:
# zUtils will look for diagnostics directory bitstreams in the
#$ZEBU_ROOT/etc/firmwares/ZSE/dt/V6.0 directory.

##### Input system cables configuration file #####
# This input file will only be taken into account with a 5Sx multi-
unit system configuration.
# "cables_xU.dff" is the input file which defines inter-unit system
cables connections.
# 'xU' is the number of units in the ZeBu Server system.
# This file is present in the release $ZEBU_ROOT/etc/
configurations.
# The generated config.dff file (see above) will contain a copy of
this file.
# Example for a '2-unit' ZeBu Server system configuration
#$systemCableFile = "/zebu/release/etc/configurations/
cables_2U.dff";

# NOTE: if $systemCableFile is not specified:
# zUtils will look for the input system cables configuration file
in $ZEBU_ROOT/etc/configurations/cables_xU.dff which is compatible
with patchFile

#####
##### MISC. DECLARATIONS #####
#####

# Do not launch memories tests at the end of setup:
#$noMemTest = 1;

```



```
# Copy the diagnostics bitstreams directory instead of the link:
#$copyBitstreams = 1;

# Do not launch zConfig at the end of setup:
#$zConfig = 0;

# Do not use $ZEBU_SYSTEM_DIR/history (faulty pins history)
#$noHistory = 1;

# Specify default Unit #num Smarti Z-ICE Vcc (1.5 V by default)
# See below for available power supply voltages (e.g. Unit 0)
# Select one of the following lines:
#$U0.SmartZice.Vcc = "1.5 V";
#$U0.SmartZice.Vcc = "1.8 V";
#$U0.SmartZice.Vcc = "2.5 V";
#$U0.SmartZice.Vcc = "3.3 V";

# Specify default Unit #num Module #num 8C-ICE Vcc Ios #num, 1.2 V
by default on ZS3 (Virtex7)
# See below for available power supply voltages (e.g. Unit 0 Module
0 VccIo 1 and 2)
# Select one lines for each Vcc per 8C-ICE Module:
#$U0.M0.DirectIce.VccIo_1 = "1.2 V";
#$U0.M0.DirectIce.VccIo_1 = "1.35 V";
```

```
#U0.M0.DirectIce.VccIo_1 = "1.5 V";  
#U0.M0.Directice.VccIo_1 = "1.8 V";  
#U0.M0.DirectIce.VccIo_2 = "1.2 V";  
#U0.M0.DirectIce.VccIo_2 = "1.5 V";  
#U0.M0.DirectIce.VccIo_2 = "1.35 V";  
#U0.M0.Directice.VccIo_2 = "1.8 V";
```