



Imperas Guide to using Virtual Platforms

Platform Specific Information for
mips.ovpworld.org / MipsMaltaLinux_TLM2.0

Imperas Software Limited

Imperas Buildings, North Weston
Thame, Oxfordshire, OX9 2HA, U.K.
docs@imperas.com.



Author	Imperas Software Limited
Version	20150901.0
Filename	Imperas_Platform_User_Guide_MipsMaltaLinux_TLM2.0.pdf
Created	26 August 2015
Status	OVP Standard Release

Copyright Notice

Copyright 2015 Imperas Software Limited. All rights reserved. This software and documentation contain information that is the property of Imperas Software Limited. The software and documentation are furnished under a license agreement and may be used or copied only in accordance with the terms of the license agreement. No part of the software and documentation may be reproduced, transmitted, or translated, in any form or by any means, electronic, mechanical, manual, optical, or otherwise, without prior written permission of Imperas Software Limited, or as expressly provided by the license agreement.

Right to Copy Documentation

The license agreement with Imperas permits licensee to make copies of the documentation for its internal use only. Each copy shall include all copyrights, trademarks, service marks, and proprietary rights notices, if any.

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 readers responsibility to determine the applicable regulations and to comply with them.

Disclaimer

IMPERAS SOFTWARE LIMITED, 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.

Model Release Status

This model is released as part of OVP releases and is included in OVPworld packages. Please visit OVPworld.org.

Table Of Contents

1.0 Virtual Platform: MipsMaltaLinux_TLM2.0	5
1.1 Licensing	5
1.2 Description	5
1.3 Limitations	5
1.4 Reference	5
1.5 Location	5
2.0 Processor [mips.ovpworld.org/processor/mips32/1.0] instance: mipsle1	6
2.1 Processor model type: 'mips32' variant '24KEf' definition	6
2.2 Instance Parameters	6
2.3 Memory Map for processor 'mipsle1' bus: 'bus1'	7
2.4 Net Connections to processor: 'mipsle1'	8
3.0 Peripheral Instances	8
3.1 Peripheral [mips.ovpworld.org/peripheral/SmartLoaderLinux/1.0] instance: Core_Board_SDRAM_promInit	8
3.2 Peripheral [intel.ovpworld.org/peripheral/82371EB/1.0] instance: PIIX4	9
3.3 Peripheral [intel.ovpworld.org/peripheral/PciIDE/1.0] instance: PIIX4-IDE	9
3.4 Peripheral [intel.ovpworld.org/peripheral/PciUSB/1.0] instance: PCI_USB	9
3.5 Peripheral [intel.ovpworld.org/peripheral/PciPM/1.0] instance: PCI_PM	10
3.6 Peripheral [amd.ovpworld.org/peripheral/79C970/1.0] instance: PCI_NET	10
3.7 Peripheral [intel.ovpworld.org/peripheral/8259A/1.0] instance: intCtrlMaster	11
3.8 Peripheral [intel.ovpworld.org/peripheral/8259A/1.0] instance: intCtrlSlave	11
3.9 Peripheral [intel.ovpworld.org/peripheral/Ps2Control/1.0] instance: Ps2	12
3.10 Peripheral [intel.ovpworld.org/peripheral/8253/1.0] instance: pit	12
3.11 Peripheral [motorola.ovpworld.org/peripheral/MC146818/1.0] instance: rtc	13
3.12 Peripheral [national.ovpworld.org/peripheral/16550/1.0] instance: uartTTY0	13
3.13 Peripheral [national.ovpworld.org/peripheral/16550/1.0] instance: uartTTY1	14
3.14 Peripheral [mips.ovpworld.org/peripheral/16450C/1.0] instance: uartCBUS	15
3.15 Peripheral [intel.ovpworld.org/peripheral/82077AA/1.0] instance: fd0	15
3.16 Peripheral [mips.ovpworld.org/peripheral/MaltaFPGA/1.0] instance: maltaFpga	16
3.17 Peripheral [ovpworld.org/peripheral/Alpha2x16Display/1.0] instance: alphaDisplay	16
3.18 Peripheral [marvell.ovpworld.org/peripheral/GT6412x/1.0] instance: sysControl	17
4.0 Overview of Imperas OVP Virtual Platforms	18
5.0 Getting Started with Imperas OVP Virtual Platforms	19
6.0 Simulating Software	19
6.1 Getting a license key to run	19
6.2 Normal runs	19
6.3 Loading Software	19
6.4 Semihosting	20
6.5 Using a terminal (UART)	20
6.6 Interacting with the simulation (keyboard and mouse)	20

6.7 More Information (Documentation) on Simulation	20
7.0 Debugging Software running on an Imperas OVP Virtual Platform	20
7.1 Debugging with GDB	20
7.2 Debugging with Imperas M*DBG	21
7.3 Debugging with the Imperas iGui and GDB	21
7.4 Debugging with the Imperas iGui and M*DBG	21
7.5 Debugging with Eclipse	21
7.6 Debugging applications running under a simulated operating system	21
8.0 Modifying the Platform	22
8.1 Platforms use C/C++ and OVP APIs	22
8.2 Platforms/Peripherals can be easily built with iGen from Imperas	22
8.3 Re-configuring the platform	22
8.4 Replacing peripherals components	23
8.5 Adding new peripherals components	23
9.0 Available Virtual Platforms	24

1.0 Virtual Platform: MipsMaltaLinux_TLM2.0

This document provides the details of the usage of an Imperas OVP Virtual Platform. The first half of the document covers specifics of this particular virtual platform. For more information about Imperas OVP virtual platforms, how they are built and used, please see the later sections in this document.

1.1 Licensing

Open Source Apache 2.0

1.2 Description

This is a platform representing a MIPS Malta development board.

It provides the peripherals required to boot and run a Linux Operating System.

A single MIPS32 architecture processor is instantiated in this platform.

This instance could be duplicated to instantiate further processors to easily create an SMP platform.

Attributes are provided for configuration of the generic ISA model for a specific processor.

The processor model is configured to operate as a MIPS32 4KEc.

The main SDRAM and Flash memory is modeled using RAM models. Both are initialised in places by the

'SmartLoaderLinux'. The SmartLoaderLinux allows ease of use of changing kernel command lines, loading an initial ram disk and creating the boot flash(s). The operation of the SmartloaderLinux is configured

using a number of attributes.

The kernel attribute of the SmartLoaderLinux and the imagefile of the processor must be consistent.

NOTE: a non Mips Malta peripheral 'AlphaDisplay16x2' has been defined in this platform definition to be used for demo purposes. It should be removed if there is a memory error in the address space 0x18000100-0x18000103

If this platform is not part of your installation, then it is available for download from www.OVPworld.org/MIPSuser.

1.3 Limitations

Verification has only been carried out using Little Endian memory ordering.

1.4 Reference

MIPS Malta User's Manual MD00048-2B-MALTA-USM-1.07.pdf

CoreFPGA User's Manual MD00116-2B-COREFPGA-USM-01.00.pdf

Linux for the MIPS Malta Development Platform Useer's Guide MD00646-2B-LINUXMALTA-USM-01.03.pdf

1.5 Location

The MipsMaltaLinux_TLM2.0 virtual platform is located in an Imperas/OVP installation at the VLNV: [mips.ovpworld.org / platform / MipsMaltaLinux_TLM2.0 / 1.0](https://mips.ovpworld.org/platform/MipsMaltaLinux_TLM2.0/1.0).

2.0 Processor [mips.ovpworld.org/processor/mips32/1.0] instance: mipsle1

2.1 Processor model type: 'mips32' variant '24KEf' definition

Imperas OVP processor models support multiple variants and details of the variants implemented in this model can be found in:

- the Imperas installation located at ImperasLib/source/mips.ovpworld.org/processor/mips32/1.0/doc
- the OVP website: [OVP Model Specific Information mips32 24KEf.pdf](https://mips.ovpworld.org/processor/mips32/1.0/doc/OVP_Model_Specific_Information_mips32_24KEf.pdf)

2.1.1 Description

MIPS32 Configurable Processor Model

2.1.2 Licensing

Usage of binary model under license governing simulator usage. Source of model available under Imperas Software License Agreement.

2.1.3 Limitations

If this model is not part of your installation, then it is available for download from www.OVPworld.org/MIPSuser.

2.1.4 Verification

Models have been validated correct as part of the MIPS Verified program and run through the MIPS AVP test programs

2.1.5 Features

MIPS32 Instruction set implemented

MMU Type: Standard TLB

FPU implemented

L1 I and D cache model in either full or tag-only mode implemented (disabled by default)

Vectored interrupts implemented

MIPS16e ASE implemented

DSP ASE implemented

2.2 Instance Parameters

Several parameters can be specified when a processor is instanced in a platform. For this processor instance 'mipsle1' it has been instanced with the following parameters:

Table 1. Processor Instance 'mipsle1' Parameters (Configurations)

Parameter	Value	Description
endian	little	Select processor endian (big or little)
simulateexceptions	simulateexceptions	Causes the processor simulate exceptions instead of

		halting
mips	100	The nominal MIPS for the processor
startaddress	0xbfc00000	Set this to override the image file start address or physical reset address

Table 2. Processor Instance 'mipsle1' Parameters (Attributes)

Parameter Name	Value	Type
variant	24KEf	
vectoredinterrupt	disable	
config1MMUSizeM1	63	

2.3 Memory Map for processor 'mipsle1' bus: 'bus1'

Processor instance 'mipsle1' is connected to bus 'bus1' using master port 'INSTRUCTION'.

Processor instance 'mipsle1' is connected to bus 'bus1' using master port 'DATA'.

Table 3. Memory Map ('mipsle1' / 'bus1' [width: 32])

Lo Address	Hi Address	Instance	Component
0x0	0x7FFFFFFF	Core_Board_SDRAM	ram
remappable	remappable	PCI_NET	79C970
remappable	remappable	PCI_PM	PciPM
remappable	remappable	PCI_USB	PciUSB
remappable	remappable	PIIX4-IDE	PciIDE
remappable	remappable	sysControl	GT6412x
0x18000020	0x18000021	intCtrlMaster	8259A
0x18000040	0x18000043	pit	8253
0x18000060	0x18000067	Ps2	Ps2Control
0x18000070	0x18000071	rtc	MC146818
0x180000A0	0x180000A1	intCtrlSlave	8259A
0x18000100	0x18000103	alphaDisplay	Alpha2x16Display
0x180002F8	0x180002FF	uartTTY1	16550
0x180003F0	0x180003F7	fd0	82077AA
0x180003F8	0x180003FF	uartTTY0	16550
0x180004D0	0x180004D0	intCtrlMaster	8259A
0x180004D1	0x180004D1	intCtrlSlave	8259A
0x1E000000	0x1E0FFFFFFF	map	bridge
0x1F000000	0x1F0008FF	maltaFpga	MaltaFPGA
0x1F000900	0x1F00093F	uartCBUS	16450C
0x1F000A00	0x1F000FFF	maltaFpga	MaltaFPGA
0x1FC00000	0x1FC0000F	remap1	bridge
0x1FC00010	0x1FC00013	Core_Board_SDRAM_promInit	SmartLoaderLinux
0x1FC00014	0x1FFFFFFF	remap2	bridge

Table 4. Bridged Memory Map ('mipsle1' / 'map' / 'flashBus' [width: 32])

Lo Address	Hi Address	Instance	Component
0x1E000000	0x1E3FFFFFFF	Monitor_Flash	ram

Table 5. Bridged Memory Map ('mipsle1' / 'remap1' / 'flashBus' [width: 32])

Lo Address	Hi Address	Instance	Component
0x1E000000	0x1E3FFFFFF	Monitor_Flash	ram

Table 6. Bridged Memory Map ('mipsle1' / 'remap2' / 'flashBus' [width: 32])

Lo Address	Hi Address	Instance	Component
0x1E000000	0x1E3FFFFFF	Monitor_Flash	ram

2.4 Net Connections to processor: 'mipsle1'

Table 7. Processor Net Connections ('mipsle1')

Net Port	Net	Instance	Component
hwint0	i8259Int	intCtrlMaster	8259A

3.0 Peripheral Instances

3.1 Peripheral [mips.ovpworld.org/peripheral/SmartLoaderLinux/1.0] instance: *Core_Board_SDRAM_promInit*

3.1.1 Description

Smart peripheral creates memory initialization for a MIPS32 based Linux kernel boot.

Performs the generation of boot code at the reset vector (virtual address 0xbfc00000) of the MIPS32 processor.

Loads both the Linux kernel and initial ramdisk into memory and patches the boot code to jump to the kernel start. Initializes the MIPS32 registers and Linux command line

3.1.2 Licensing

Open Source Apache 2.0

3.1.3 Limitations

None

3.1.4 Reference

MIPS Malta User Manual. MIPS Boot code reference

Table 8. Configuration options (attributes) set for instance 'Core_Board_SDRAM_promInit'

Attributes	Value
kernel	vmlinux
boardid	0x00000420
initrd	initrd.gz
root	root
nonelinux	nonelinux
bootimage	bootimage

command	console=ttyS0
---------	---------------

3.2 Peripheral [intel.ovpworld.org/peripheral/82371EB/1.0] instance: PIIX4

3.2.1 Description

PIIX4 PCI configuration controller

3.2.2 Licensing

Open Source Apache 2.0

3.2.3 Limitations

This model has sufficient functionality to allow a Linux Kernel to Boot on the MIPS:MALTA platform.

3.2.4 Reference

Intel 82371EB South Bridge Chipset Datasheet

Table 9. Configuration options (attributes) set for instance 'PIIX4'

Attributes	Value
PCISlot	10

3.3 Peripheral [intel.ovpworld.org/peripheral/PciIDE/1.0] instance: PIIX4-IDE

3.3.1 Description

PCI:IDE interface. This forms part of the 82371 PIIX4 chip. It implements 4 IDE interfaces and 2 DMA controllers

3.3.2 Licensing

Open Source Apache 2.0

3.3.3 Limitations

This model has sufficient functionality to allow a Linux Kernel to Boot on the MIPS:MALTA platform

3.3.4 Reference

Intel 82371EB South Bridge Chipset Datasheet

Table 10. Configuration options (attributes) set for instance 'PIIX4-IDE'

Attributes	Value
PCISlot	10
PCIfunction	1
Drive0Name	mipsel_hda
Drive1Name	mipsel_hdb
Drive2Name	mipsel_cd

3.4 Peripheral [intel.ovpworld.org/peripheral/PciUSB/1.0] instance: PCI_USB

3.4.1 Description

PCI USB Interface

3.4.2 Licensing

Open Source Apache 2.0

3.4.3 Limitations

This model has sufficient functionality to allow a Linux Kernel to Boot on the MIPS:MALTA platform

3.4.4 Reference

Intel 82371EB South Bridge Chipset Datasheet

Table 11. Configuration options (attributes) set for instance 'PCI_USB'

Attributes	Value
PCIslot	10
PCIfunction	2

3.5 Peripheral [intel.ovpworld.org/peripheral/PciPM/1.0] instance: PCI_PM

3.5.1 Description

PCI Power Manager

3.5.2 Licensing

Open Source Apache 2.0

3.5.3 Limitations

This model has sufficient functionality to allow a Linux Kernel to Boot on the MIPS:MALTA platform

3.5.4 Reference

Intel 82371EB South Bridge Chipset Datasheet

Table 12. Configuration options (attributes) set for instance 'PCI_PM'

Attributes	Value
PCIslot	10
PCIfunction	3

3.6 Peripheral [amd.ovpworld.org/peripheral/79C970/1.0] instance: PCI_NET

3.6.1 Description

PCI:Ethernet interface. Implements part of the AMD AM79C97xx series devices.
diagnosticlevel:bits 0:1

give levels 0:3 for the network hardware.bits 2:3 give levels 0:3 for the user:mode SLIRP interface.

3.6.2 Licensing

Open Source Apache 2.0

3.6.3 Limitations

Sufficient to Boot MIPS Linux.

3.6.4 Reference

AMD Am79C973/Am79C975 PCnet-FAST III Single-Chip 10/100 Mbps PCI Ethernet Controller with Integrated PHY Datasheet

Table 13. Configuration options (attributes) set for instance 'PCI_NET'

Attributes	Value
PCIslot	11
PCIfunction	0
pollDelay	1000
redir	redir
tftpPrefix	tftpPrefix
ethereal	ethereal

3.7 Peripheral [intel.ovpworld.org/peripheral/8259A/1.0] instance: *intCtrlMaster*

3.7.1 Licensing

Open Source Apache 2.0

3.7.2 Description

Intel 8259A Programmable Interrupt Controller (PIT).

3.7.3 Limitations

This model has sufficient functionality to allow a Linux Kernel to Boot on the MIPS:MALTA platform.

3.7.4 Reference

Intel 8259A Datasheet. MIPS Malta Platform Reference Guide.

Table 14. Configuration options (attributes) set for instance 'intCtrlMaster'

Attributes	Value
spen	master

3.8 Peripheral [intel.ovpworld.org/peripheral/8259A/1.0] instance: *intCtrlSlave*

3.8.1 Licensing

Open Source Apache 2.0

3.8.2 Description

Intel 8259A Programmable Interrupt Controller (PIT).

3.8.3 Limitations

This model has sufficient functionality to allow a Linux Kernel to Boot on the MIPS:MALTA platform.

3.8.4 Reference

Intel 8259A Datasheet. MIPS Malta Platform Reference Guide.

Table 15. Configuration options (attributes) set for instance 'intCtrlSlave'

Attributes	Value
spen	slave

3.9 Peripheral [intel.ovpworld.org/peripheral/Ps2Control/1.0] instance: Ps2

3.9.1 Description

PS2 Keyboard/Mouse Controller

3.9.2 Licensing

Open Source Apache 2.0

3.9.3 Limitations

This is a preliminary model with sufficient functionality to enable Linux to Boot on the MIPS:MALTA platform.

Mouse functions are currently turned off

3.9.4 Reference

SMsC FDC37M817 Super I/O Controller Datasheet

Table 16. Configuration options (attributes) set for instance 'Ps2'

Attributes	Value
pollPeriod	50000
disableInput	disableInput
grabDisable	1

3.10 Peripheral [intel.ovpworld.org/peripheral/8253/1.0] instance: pit

3.10.1 Description

Intel 8253 Programmable Interval Timer (PIT)

3.10.2 Limitations

This model has sufficient functionality to allow a Linux Kernel to Boot on the MIPS:MALTA platform.

Not all modes are supported.

3.10.3 Licensing

Open Source Apache 2.0

3.10.4 Reference

Intel 8253 Datasheet. MIPS Malta Platform Reference Guide.

There are no configuration options set for this peripheral instance.

3.11 Peripheral [motorola.ovpworld.org/peripheral/MC146818/1.0] instance: rtc

3.11.1 Description

MC146818 Real:time clock

3.11.2 Licensing

Open Source Apache 2.0

3.11.3 Limitations

This model has sufficient functionality to allow a Linux Kernel to Boot on the MIPS:MALTA platform

3.11.4 Reference

Motorola MC146818AS Datasheet

Table 17. Configuration options (attributes) set for instance 'rtc'

Attributes	Value
timefromhost	timefromhost

3.12 Peripheral [national.ovpworld.org/peripheral/16550/1.0] instance: uartTTY0

3.12.1 Licensing

Open Source Apache 2.0

3.12.2 Description

16550 UART model

The serial input/output from the simulator is implemented using the Serial Device Support described in OVP BHM and PPM API Functions Reference, which describes the parameters that control how the model interacts with the host computer.

Interrupts and FIFOs are supported.

Registers are aligned on 1 byte boundaries.

3.12.3 Limitations

Resolution of the baud rate is limited to the simulation time slice (aka quantum) size.

Values written to the MCR are ignored. Loopback mode is not supported.

The LSR is read-only. The model never sets the LSR 'Parity Error', 'Framing Error', 'Break Interrupt' or 'Error in RCVR FIFO' bits.

The MSR 'Data Set Ready' and 'Clear To Send' bits are set at reset and all other MSR bits are cleared. MSR bits will only be changed by writes to the MSR and values written to the Modem Status Register do not effect the operation of the model.

3.12.4 Reference

PC16550D Universal Asynchronous Receiver/Transmitter with FIFOs datasheet
(<http://www.ti.com/lit/ds/symlink/pc16550d.pdf>)

Table 18. Configuration options (attributes) set for instance 'uartTTY0'

Attributes	Value
infile	infile
outfile	uartTTY0.log
portnum	portnum
console	enable
finishOnDisconnect	y

3.13 Peripheral [*national.ovpworld.org/peripheral/16550/1.0*] instance: *uartTTY1*

3.13.1 Licensing

Open Source Apache 2.0

3.13.2 Description

16550 UART model

The serial input/output from the simulator is implemented using the Serial Device Support described in OVP BHM and PPM API Functions Reference, which describes the parameters that control how the model interacts with the host computer.

Interrupts and FIFOs are supported.

Registers are aligned on 1 byte boundaries.

3.13.3 Limitations

Resolution of the baud rate is limited to the simulation time slice (aka quantum) size.

Values written to the MCR are ignored. Loopback mode is not supported.

The LSR is read-only. The model never sets the LSR 'Parity Error', 'Framing Error', 'Break Interrupt' or 'Error in RCVR FIFO' bits.

The MSR 'Data Set Ready' and 'Clear To Send' bits are set at reset and all other MSR bits are cleared. MSR bits will only be changed by writes to the MSR and values written to the Modem Status Register do not effect the operation of the model.

3.13.4 Reference

PC16550D Universal Asynchronous Receiver/Transmitter with FIFOs datasheet

(<http://www.ti.com/lit/ds/symlink/pc16550d.pdf>)

Table 19. Configuration options (attributes) set for instance 'uartTTY1'

Attributes	Value
infile	infile
outfile	uartTTY1.log
portnum	portnum

3.14 Peripheral [mips.ovpworld.org/peripheral/16450C/1.0] instance: uartCBUS

3.14.1 Description

Model of 16550/16450 UART.

Special version with register addresses for MIPS MALTA C-BUS.

Connects to a bus by a slave port and optionally to a processor by an interrupt signal.

The serial input/output ports are modeled by socket connection which must be attached to a process outside the simulation environment. Note that on start:up, the UART model will block the simulator, pending a connection to the socket

3.14.2 Licensing

Open Source Apache 2.0

3.14.3 Limitations

No modeling of baud:rate.

No modem support (DTR etc).

No support for parity.

No means to simulate errors

3.14.4 Reference

MIPS Malta Datasheet

Table 20. Configuration options (attributes) set for instance 'uartCBUS'

Attributes	Value
infile	infile
outfile	outfile
portnum	portnum

3.15 Peripheral [intel.ovpworld.org/peripheral/82077AA/1.0] instance: fd0

3.15.1 Description

Dummy Floppy Disc Controller.

3.15.2 Licensing

Open Source Apache 2.0

3.15.3 Limitations

Register stubs only.

3.15.4 Reference

<http://www.buchty.net/casio/files/82077.pdf> <http://www.alldatasheet.com/datesheet-pdf/pdf/167793/INTEL/82077AA.html>

There are no configuration options set for this peripheral instance.

3.16 Peripheral [mips.ovpworld.org/peripheral/MaltaFPGA/1.0] instance: *maltaFpga*

3.16.1 Description

MIPS MALTA FPGA. Drives Development board functions

3.16.2 Licensing

Open Source Apache 2.0

3.16.3 Limitations

This model has sufficient functionality to allow a Linux Kernel to Boot on the MIPS:MALTA platform

3.16.4 Reference

MIPS Malta User Manual

Table 21. Configuration options (attributes) set for instance 'maltaFpga'

Attributes	Value
stoponsoftreset	1

3.17 Peripheral [ovpworld.org/peripheral/Alpha2x16Display/1.0] instance: *alphaDisplay*

3.17.1 Description

This is a simple test peripheral creating a 2x16 alphanumeric display.

3.17.2 Licensing

Open Source Apache 2.0

3.17.3 Limitations

This is not representing a real device and provides simple operations as an example.

3.17.4 Reference

This is not based upon a real device

There are no configuration options set for this peripheral instance.

3.18 Peripheral [marvell.ovpworld.org/peripheral/GT6412x/1.0] instance: sysControl

3.18.1 Description

GT64120 System Controller.

Diagnostic levels:

PCI_SLAVE	0x03
PCI_CONFIG_MASTER	0x04
PCI_EMPTY	0x08
INT_ACK	0x10
MAIN_BUS	0x20
INFO	0x40

3.18.2 Licensing

Open Source Apache 2.0

3.18.3 Limitations

This model has sufficient functionality to allow a Linux Kernel to Boot on the MIPS:MALTA platform.

3.18.4 Reference

GT64121A System Controller for RC4650/4700/5000 and RM526X/527X/7000 CPUs Datasheet Revision 1.0 MAR 14, 2000

There are no configuration options set for this peripheral instance.

4.0 Overview of Imperas OVP Virtual Platforms

This document provides the details of the usage of an Imperas OVP Virtual Platform. The first half of the document covers specifics of this particular virtual platform.

This second part of the document, includes information about Imperas OVP virtual platforms, how they are built and used.

The Imperas virtual platforms are designed to provide a base for you to run high-speed software simulations of CPU-based SoCs and platforms on any suitable PC. They are typically based on the functionality of vendors fixed or evaluation platforms, enabling you to simulate software on these reference platforms. Typically virtual platforms are fixed and require the vendor to modify or extend them. Imperas virtual platforms are different in that they enable you to extend the functionality of the virtual platform, to closer reflect your own platform, by adding more component models, running different operating systems or adding additional applications.

Imperas virtual platforms are created using the Imperas iGen technology, allowing them to be used with Imperas OVP based simulators and also with Accellera/OSCI compliant SystemC simulators and commercial EDA System Design environments that use SystemC.

Virtual platforms include simulation models of the target devices, including the processor model(s) for the target device plus enough peripheral models to boot an operating system or run bare metal applications. The platform and the peripheral models used in most of the virtual platforms are open source, so that you can easily add new models to the platform as well as modify the existing models. Some models are only provided as binary, normally because the IP owner has restricted the release of the model source. In this case, please contact Imperas for more information.

There are typically several generic flavors of the virtual platforms for specific processor families, some targeting full operating systems, such as Linux, and some which focus on Real Time Operating Systems (RTOS) such as Mentor Nucleus or freeRTOS. OVP models of the processor cores are included in the virtual platforms, and for those processors which support multiple cores SMP Linux is often supported for that virtual platform. For all of these virtual platforms, many of the peripheral components of the platform are modeled, often including the Ethernet and USB components. The semi-hosting capability of the Imperas virtual platform simulator products enables connection via the Ethernet and USB components from the virtual platform to the real world via the x86 host machine.

The Imperas OVP CPU models are written using the OVP Virtual Machine Interface (VMI) API that defines the behavior of the processor. The VMI API makes a clear line between model and simulator allowing very good optimization and world class high speed performance. The processor models are Instruction Accurate and do not model the detailed cycle timing of the processor and they implement functionality at the level of a Programmers View of the processor and peripherals and the software running on them does not know it is not running on hardware. Many models are provided as a binary shared object

and also as source. This allows the download and use of the model binary or the use of the source to explore and modify the model. The models are run through an extensive QA and regression testing process and most processor model families are validated using technology provided by the processor IP owners. All the models in this platform are developed with the Open Virtual Platforms APIs and are implemented in C.

More information on modeling and APIs can be found on the www.OVPworld.org site.

5.0 Getting Started with Imperas OVP Virtual Platforms

Virtual platforms are downloadable from the OVPworld website OVPworld.org/downloads. You need to browse and look for '<platform processor name> Examples'. You do need to be registered and logged in on the OVP site to download. OVPworld currently provides 32 bit host versions of packages containing virtual platforms.

When downloading, choose, Linux or Windows host. 32 bit packages can be installed and executed on 32 bit or 64 bit hosts. If you require a 64 bit host version please contact Imperas.

For example, for the ARM Versatile Express platform booting Linux on Cortex-A15MP Single, Dual, and Quad core processors, you would want the download package:
'OVPSim_demo_Linux_ArmVersatileExpress_arm_Cortex-A15MP'.

Most virtual platform packages contain the platform and all the processor and peripheral models needed. You will need to download a simulator to run the platform. You can use OVPSim, downloadable from OVPworld.org/downloads, or you can use one of the Imperas simulators (imperas.com/products) available commercially from Imperas.

6.0 Simulating Software

6.1 Getting a license key to run

After you have downloaded you will need a runtime license key before the simulators will run. For OVPSim please visit OVPworld.org/likey and provide the required information and an evaluation/demo license key will be automatically sent to you. If you are using Imperas, then please contact Imperas for a license key.

6.2 Normal runs

To run a platform, read the section below on command line control of the platform and the section on setting command line arguments.

6.3 Loading Software

For most virtual platforms the platform is already configured to run the default software application/program and there is normally a script to run that sets some arguments. You can then copy/edit this script to select your own applications etc.

The example application programs are typically .elf format files and are provided pre-compiled. There are

normally makefiles and associated scripts to recompile the example applications.

To find more information about compiling and loading software, the following document should be looked at: [Imperas Installation and Getting Started.pdf](#).

6.4 Semihosting

In a virtual platform, semihosting is not normally used as there is normally hardware that implements the appropriate functionality - for example I/O will be handled by UARTs etc.

6.5 Using a terminal (UART)

If the platform includes one or more UARTs you will need to connect a terminal program to it so that you can see output and type into the simulated program. Review the list of peripherals below and see what configuration options it has been set with. In most cases there is an option to set to instruct the simulator to 'pop up' a terminal window connected to the simulated UART.

6.6 Interacting with the simulation (keyboard and mouse)

If the platform has a simulated UART you can normally set a command to get the simulator to pop up a terminal window allowing you to see output from the simulated UART and also allowing you to type characters into the UART that can be processed by the simulated software.

If your simulated platform has an LCD device then you can often configure it to recognize mouse movements and mouse clicks - allowing full interaction.

To see these interactions in action, have a look at some of the available videos available at OVPworld.org/demosandvideos.

6.7 More Information (Documentation) on Simulation

To find more information about running simulations and more of the options the simulators provide, the following documents should be looked at:

[Imperas Installation and Getting Started.pdf](#)

[OVPsim and CpuManager User Guide.pdf](#)

[OVP Control File User Guide.pdf](#)

A full list of the currently available OVP documentation is available: OVPworld.org/documentation.

7.0 Debugging Software running on an Imperas OVP Virtual Platform

The Imperas and OVP simulators have several different interfaces to debuggers. These include several proprietary formats and also the standard GNU RSP format is supported allowing many compatible debuggers to be used. Below are some examples that Imperas directly support.

7.1 Debugging with GDB

A GNU debugger (GDB) can be connected to a processor in a platform using the RSP protocol. This allows

the application program running on a processor to be debugged using a specific GDB for the processor selected. When using the Imperas Professional products many connections can be made allowing a GDB to be connected to all the processors in the platform.

The use of GDB is documented: [OVPSim Debugging Applications with GDB User Guide.pdf](#).

7.2 Debugging with Imperas M*DBG

The Imperas multi-processor debugger can be connected to a platform and through this connection you can debug application programs running on all of the processors instanced within the platform. It is also capable, within this single unified environment, to debug peripheral model behavioral code in conjunction with the processor application programs.

For more information please see the Imperas M*DBG user guide.

The Imperas multi-processor debugger is also capable of controlling the Imperas Verification Analysis and Profiling (VAP) tools in real time, making them invaluable to application program development, debugging and analysis.

For more information please see the Imperas VAP tools user guide.

7.3 Debugging with the Imperas iGui and GDB

Imperas iGui gives a GUI front end to the use of the GDB debugger. It allows use of all the features of GDB including source level application program debugging on processors.

7.4 Debugging with the Imperas iGui and M*DBG

Imperas iGui gives a GUI front end to the Imperas multi-processor debugger. It provides all the features of this debugger but does so with source level application program debugging on processors and source level debugging of the behavioral code on peripheral components in the platform. A context view shows all the processor and peripheral components within the platform and allows switching between them to examine the state of each at the event at which the simulation was stopped

Imperas iGui provides a menu from which the Imperas VAP tools can be controlled.

7.5 Debugging with Eclipse

A standard Eclipse CDT development environment can be connected to one or more processors in a platform (multiple processors require an Imperas professional product). The simulation platform is started remotely or using the external tool feature in Eclipse, opens a debug port and awaits the connection with Eclipse. All features provided by the Eclipse CDT development environment are available to be used to debug software applications executing on the processors in the platform.

The use of Eclipse is documented: [OVPSim Debugging Applications with Eclipse User Guide.pdf](#).

7.6 Debugging applications running under a simulated operating system

If the simulated platform is running an Operating System and the platform has a UART or Ethernet etc connection then it is often possible to connect an external debugger and debug the applications running under the simulated operating system.

An example would be a simulated platform running the Linux operating system, such as the MIPS Malta, or ARM Versatile Express. Within the simulated Linux you can start a gdbserver that connects from within the simulation through a UART out to the host PC via a port. Within the host PC you start a terminal program and connect to the port with a debugger such as GDB and can then debug the simulated user application.

8.0 Modifying the Platform

8.1 Platforms use C/C++ and OVP APIs

The Imperas and OVP simulators execute a platform that is written in C/C++ and that makes function calls into the simulator's APIs. Thus the virtual platform is compiled from C/C++ into a binary shared object that the simulator loads and runs. OVP provides the definition and documentation that defines the C APIs for modeling the platforms, the peripherals and the processors. You can find more information about these APIs on the OVP website and in the OVP API documentation.

8.2 Platforms/Peripherals can be easily built with iGen from Imperas

Imperas provides a product 'iGen' that takes an input script file and creates the C/C++ files needed for platforms and peripherals - it creates the C/C++ file that is compiled into the platform or peripheral that is needed as an object file by the simulator. iGen creates the C/C++ files, you then need to add any necessary behaviors or further details etc. For platforms iGen creates either a C platform or a C++ SystemC TLM2 platform. For peripherals iGen creates the C files and also provides a native C++ SystemC TLM2 interface to allow the peripheral to be instantiated in SystemC TLM2 platforms.

Information on iGen is available from: imperas.com/products.

8.3 Re-configuring the platform

There will normally be several configuration options that you can set when running the platform without the need to change any source. Refer to the section above on command line arguments. If these do not allow you to make the changes you need, then you may need to edit and recompile the source of the platform.

The source of the platform and the source of the peripherals will be installed as part of the packages you are using. The sources are located in the Imperas/OVP installation VLNV source tree. The VLNV term refers to: Vendor (eg arm.ovpworld.org), Library (eg platform), Name, (eg ArmVersatileExpress-CA15), and Version (eg 1.0). To modify the platform, locate the platform source files.

If you are an Imperas user and have access to iGen, we recommend you modify the source script files and regenerate and recompile the C that makes up the platform. Refer to the Imperas iGen model generator

guide and the Imperas platform generator guide.

If you are using the C or SystemC TLM2 platforms with OVPsim, then you can edit the C/C++ files, recompile the source directly using the supplied makefiles, and then run the simulator directly with the resultant shared object.

8.4 Replacing peripherals components

If you need to replace peripherals, find the appropriate place in the source of the platform, make the change you need, and recompile etc. Look in the library for documentation on available peripherals and their configuration options.

8.5 Adding new peripherals components

If you need to add peripherals, find the appropriate place in the source, make the additions you need, and recompile etc. Look in the library for documentation on available peripherals and their configuration options.

If you need to create new peripheral components then use iGen to very quickly create the necessary C/C++ files that get you started. With iGen you can create peripherals with register/memory state in a few lines of iGen source. When adding behavior to the peripherals refer to the OVP API documentation.

9.0 Available Virtual Platforms

Table 22. Imperas / OVP Extendable Platform Kits (17 available)

Platform Name	Vendor
AlteraCycloneIII_3c120	altera.ovpworld.org
AlteraCycloneV_HPS	altera.ovpworld.org
AlteraCycloneV_HPS_TLM2	altera.ovpworld.org
ARMv8-A-FMv1	arm.ovpworld.org
ArmIntegratorCP	arm.ovpworld.org
ArmIntegratorCP_TLM2.0	arm.ovpworld.org
ArmVersatileExpress	arm.ovpworld.org
ArmVersatileExpress-CA15	arm.ovpworld.org
ArmVersatileExpress-CA9	arm.ovpworld.org
ArmVersatileExpress_CA9_TLM2	arm.ovpworld.org
AtmelAT91SAM7	atmel.ovpworld.org
FreescaleKinetis60	freescale.ovpworld.org
FreescaleVybridVF5	freescale.ovpworld.org
MipsMalta	mips.ovpworld.org
MipsMaltaLinux_TLM2.0	mips.ovpworld.org
RenesasUPD70F3441	renesas.ovpworld.org
XilinxML505	xilinx.ovpworld.org

Table 23. Imperas General Virtual Platforms (6 available)

Platform Name	Vendor
arm-ti-eabi	arm.imperas.com
armm-ti-coff	arm.imperas.com
armm-ti-eabi	arm.imperas.com
HeteroAlteraCycloneV_HPS_CycloneIII_3c120	imperas.ovpworld.org
HeteroArmNucleusMIPSLinux	imperas.ovpworld.org
QuadArmVersatileExpress	imperas.ovpworld.org

Table 24. Imperas / OVP Bare Metal Virtual Platforms (39 available)

Platform Name	Vendor
BareMetalNios_IISingle	altera.ovpworld.org
BareMetalNios_IISingle_TLM2.0	altera.ovpworld.org
BareMetalArcSingle	arc.ovpworld.org
BareMetalArcSingle_TLM2.0	arc.ovpworld.org
BareMetalArm7Single	arm.ovpworld.org
BareMetalArm7Single_TLM2.0	arm.ovpworld.org
BareMetalArmAArch64Single_TLM2.0	arm.ovpworld.org
BareMetalArmCortexADual	arm.ovpworld.org
BareMetalArmCortexASingle	arm.ovpworld.org
BareMetalArmCortexASingleAngelTrap	arm.ovpworld.org
BareMetalArmCortexASingle_TLM2.0	arm.ovpworld.org
BareMetalArmCortexMSingle	arm.ovpworld.org
BareMetalArmCortexMSingle_TLM2.0	arm.ovpworld.org

ArmCortexMFreeRTOS	imperas.ovpworld.org
ArmCortexMuCOS-II	imperas.ovpworld.org
BareMetalArcManycore24_TLM2.0	imperas.ovpworld.org
BareMetalArm7Dual_TLM2.0	imperas.ovpworld.org
BareMetalArmx1Mips32x3	imperas.ovpworld.org
BareMetalMips32Multicore2_TLM2.0	imperas.ovpworld.org
Or1kUclinux_TLM2.0	imperas.ovpworld.org
BareMetalM14KSingle	mips.ovpworld.org
BareMetalM14KSingle_TLM2.0	mips.ovpworld.org
BareMetalMips32Dual	mips.ovpworld.org
BareMetalMips32Single	mips.ovpworld.org
BareMetalMips32Single_TLM2.0	mips.ovpworld.org
BareMetalMips64Single	mips.ovpworld.org
BareMetalMips64Single_TLM2.0	mips.ovpworld.org
BareMetalMipsDual	mips.ovpworld.org
BareMetalMipsSingle	mips.ovpworld.org
BareMetalMipsSingle_TLM2.0	mips.ovpworld.org
BareMetalOr1kSingle	ovpworld.org
BareMetalOr1kSingle_TLM2.0	ovpworld.org
BareMetalM16cSingle	posedgesoft.ovpworld.org
BareMetalPowerPc32Single	power.ovpworld.org
BareMetalPowerPc32Single_TLM2.0	power.ovpworld.org
BareMetalV850Single	renesas.ovpworld.org
BareMetalV850Single_TLM2.0	renesas.ovpworld.org
ghs-multi	renesas.ovpworld.org
BareMetalMicroBlazeSingle_TLM2.0	xilinx.ovpworld.org

#