Imperas Peripheral Model Guide

Model Specific Information for mips.ovpworld.org / 16450C

Imperas Software Limited

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

Author	Imperas Software Limited
Version	20150901.0
Filename	OVP_Peripheral_Specific_Information_16450C.pdf
Created	26 August 2015
Status	OVP Standard Release

Copyright Notice

Right to Copy Documentation

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

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 Model Specific Information

This document provides usage information for an Imperas OVP peripheral behavioral model.

The document is split into sections providing specific information for this peripheral, including any ports for connecting into a platform, registers, other component parts, and configuration options and general information for peripheral modeling with Imperas OVP.

1.1 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

1.2 Licensing

Open Source Apache 2.0

1.3 Limitations

No modeling of baud:rate.

No modem support (DTR etc).

No support for parity.

No means to simulate errors

1.4 Reference

MIPS Malta Datasheet

1.5 Location

The 16450C peripheral model is located in an Imperas/OVP installation at the VLNV: mips.ovpworld.org / peripheral / 16450C / 1.0.

2.0 Peripheral Instance Parameters

This model accepts the following parameters:

Table 1. Peripheral Parameters

Name	Type	Description
console	bool	Standard Serial Socket Parameter: See OVP BHM and PPM API Function Reference: Enable automatic console
portnum	uns32	Standard Serial Socket Parameter: See OVP BHM and PPM API Function Reference: Specify port to open for a connection. A value of zero causes the OS to select the next available port.
infile	string	Standard Serial Socket Parameter: See OVP BHM and PPM API Function Reference: UART takes input from this serial input source file
outfile	string	Standard Serial Socket Parameter: See OVP BHM and PPM API Function Reference: Serial output file
portFile	string	Standard Serial Socket Parameter: See OVP BHM and PPM API Function Reference: When portnum is set to zero, write the assigned port number to this file
log	bool	Standard Serial Socket Parameter: See OVP BHM and PPM API Function Reference: Report serial output in the simulator log
finishOnDisconnect	bool	Standard Serial Socket Parameter: See OVP BHM and PPM API Function Reference: When defined the simulation will be terminated if the port is disconnected
simulatebaud	bool	If true, transmit at the programmed baud rate (the default is to transmit without any delay).
defaultbaud	uns32	Set the default baud rate multiplier. Default is 9.
charmode	bool	Puts the telnet server into char mode Default is line mode
record	string	Enable record model
replay	string	Enable replay mode
uart16550	bool	Enable 16550 mode (with FIFOS)

3.0 Net Ports

This model has the following net ports:

Table 2. Net Ports

Name	Type	Must Be Connected	Description
intOut	output	F (False)	Intetrupt output.

4.0 Bus Slave Ports

This model has the following bus slave ports:

4.1 Bus Slave Port: bport1

Table 3. Bus Slave Port: bport1

Name	Size (bytes)	Must Be Connected	Description
bport1	0x40	T (True)	Byte:wide access to control and status registers.

Table 4. Bus Slave Port: bport1 Registers:

Name	Offset	Width (bits)	Description	R/W	is Volatile

rbr	0x0	8	receive byte		
tbr	0x0	8	transmit byte		
ier	0x8	8	interrupt enable		
iir	0x10	8	interrupt input		
lcr	0x18	8	line control		
mcr	0x20	8	modem control		
lsr	0x28	8	line status		
msr	0x30	8	modem status		
scr	0x38	8	status/control		

5.0 Platforms that use this peripheral component

Peripheral components can be used in many different platforms, including those developed by Imperas or by other users of OVP. You can use this peripheral in your own platforms.

Table 5. Publicly available platforms using peripheral '16450C'

Platform Name	Vendor
HeteroArmNucleusMIPSLinux	imperas.ovpworld.org
MipsMalta	mips.ovpworld.org
MipsMaltaLinux_TLM2.0	mips.ovpworld.org

6.0 Peripheral components in the library

Table 6. Publicly available Imperas/OVP peripheral models (158 models)

Peripheral	Peripheral	Peripheral
mips.ovpworld.org/MaltaFPGA	mips.ovpworld.org/SmartLoaderLinux	motorola.ovpworld.org/MC146818
national.ovpworld.org/16450	national.ovpworld.org/16550	ovpworld.org/Alpha2x16Display
ovpworld.org/dummyPort	ovpworld.org/DynamicBridge	ovpworld.org/FlashDevice
ovpworld.org/ledRegister	ovpworld.org/SerInt	ovpworld.org/SimpleDma
ovpworld.org/VirtioBlkMMIO	philips.ovpworld.org/ISP1761	renesas.ovpworld.org/adc
renesas.ovpworld.org/bcu	renesas.ovpworld.org/brg	renesas.ovpworld.org/can
renesas.ovpworld.org/can	renesas.ovpworld.org/clkgen	renesas.ovpworld.org/crc
renesas.ovpworld.org/csib	renesas.ovpworld.org/csie	renesas.ovpworld.org/dma
renesas.ovpworld.org/intc	renesas.ovpworld.org/memc	renesas.ovpworld.org/rng
renesas.ovpworld.org/taa	renesas.ovpworld.org/tms	renesas.ovpworld.org/tmt
renesas.ovpworld.org/uartc	renesas.ovpworld.org/UPD70F3441Logic	smsc.ovpworld.org/LAN9118
smsc.ovpworld.org/LAN91C111	ti.ovpworld.org/UartInterface	xilinx.ovpworld.org/mdm
xilinx.ovpworld.org/mpmc	xilinx.ovpworld.org/xps-gpio	xilinx.ovpworld.org/xps-iic
xilinx.ovpworld.org/xps-intc	xilinx.ovpworld.org/xps-ll-temac	xilinx.ovpworld.org/xps-mch-emc
xilinx.ovpworld.org/xps-sysace	xilinx.ovpworld.org/xps-timer	xilinx.ovpworld.org/xps-uartlite
altera.ovpworld.org/dw-apb-timer	altera.ovpworld.org/dw-apb-uart	altera.ovpworld.org/IntervalTimer32Core
altera.ovpworld.org/IntervalTimer64Core	altera.ovpworld.org/JtagUart	altera.ovpworld.org/PerformanceCounterCore
altera.ovpworld.org/RSTMGR	altera.ovpworld.org/SystemIDCore	altera.ovpworld.org/Uart
amd.ovpworld.org/79C970	arm.ovpworld.org/AaciPL041	arm.ovpworld.org/CompactFlashRegs
arm.ovpworld.org/CoreModule9x6	arm.ovpworld.org/DebugLedAndDipSwitch	arm.ovpworld.org/DMemCtrlPL341
arm.ovpworld.org/IcpControl	arm.ovpworld.org/IcpCounterTimer	arm.ovpworld.org/IntICP
arm.ovpworld.org/IntICP	arm.ovpworld.org/KbPL050	arm.ovpworld.org/L2CachePL310
arm.ovpworld.org/LcdPL110	arm.ovpworld.org/MmciPL181	arm.ovpworld.org/RtcPL031
arm.ovpworld.org/SerBusDviRegs	arm.ovpworld.org/SmartLoaderArm64Linux	arm.ovpworld.org/SmartLoaderArmLinux
arm.ovpworld.org/SMemCtrlPL354	arm.ovpworld.org/SysCtrlSP810	arm.ovpworld.org/TimerSP804
arm.ovpworld.org/TzpcBP147	arm.ovpworld.org/UartPL011	arm.ovpworld.org/VexpressSysRegs
arm.ovpworld.org/WdtSP805	atmel.ovpworld.org/AdvancedInterruptController	atmel.ovpworld.org/ParallelIOController
atmel.ovpworld.org/PowerSaving	atmel.ovpworld.org/SpecialFunction	atmel.ovpworld.org/TimerCounter
atmel.ovpworld.org/UsartInterface	atmel.ovpworld.org/WatchdogTimer	cirrus.ovpworld.org/GD5446
freescale.ovpworld.org/KinetisADC	freescale.ovpworld.org/KinetisAIPS	freescale.ovpworld.org/KinetisAXBS
freescale.ovpworld.org/KinetisCAN	freescale.ovpworld.org/KinetisCMP	freescale.ovpworld.org/KinetisCMT
freescale.ovpworld.org/KinetisCRC	freescale.ovpworld.org/KinetisDAC	freescale.ovpworld.org/KinetisDDR
freescale.ovpworld.org/KinetisDMA	freescale.ovpworld.org/KinetisDMAC	freescale.ovpworld.org/KinetisDMAMUX
freescale.ovpworld.org/KinetisENET	freescale.ovpworld.org/KinetisEWM	freescale.ovpworld.org/KinetisFB
freescale.ovpworld.org/KinetisFMC	freescale.ovpworld.org/KinetisFTFE	freescale.ovpworld.org/KinetisFTM
freescale.ovpworld.org/KinetisGPIO	freescale.ovpworld.org/KinetisI2C	freescale.ovpworld.org/KinetisI2S
freescale.ovpworld.org/KinetisLLWU	freescale.ovpworld.org/KinetisLPTMR	freescale.ovpworld.org/KinetisMCG
freescale.ovpworld.org/KinetisMPU	freescale.ovpworld.org/KinetisNFC	freescale.ovpworld.org/KinetisOSC
freescale.ovpworld.org/KinetisPDB	freescale.ovpworld.org/KinetisPIT	freescale.ovpworld.org/KinetisPMC
freescale.ovpworld.org/KinetisPORT	freescale.ovpworld.org/KinetisRCM	freescale.ovpworld.org/KinetisRFSYS
freescale.ovpworld.org/KinetisRFVBAT	freescale.ovpworld.org/KinetisRNG	freescale.ovpworld.org/KinetisRTC
freescale.ovpworld.org/KinetisSDHC	freescale.ovpworld.org/KinetisSIM	freescale.ovpworld.org/KinetisSMC
freescale.ovpworld.org/KinetisSPI	freescale.ovpworld.org/KinetisTSI	freescale.ovpworld.org/KinetisUART

freescale.ovpworld.org/KinetisUSB	freescale.ovpworld.org/KinetisUSBDCD	freescale.ovpworld.org/KinetisUSBHS
freescale.ovpworld.org/KinetisVREF	freescale.ovpworld.org/KinetisWDOG	freescale.ovpworld.org/Uart
freescale.ovpworld.org/VybridADC	freescale.ovpworld.org/VybridANADIG	freescale.ovpworld.org/VybridCCM
freescale.ovpworld.org/VybridDMA	freescale.ovpworld.org/VybridGPIO	freescale.ovpworld.org/VybridI2C
freescale.ovpworld.org/VybridLCD	freescale.ovpworld.org/VybridQUADSPI	freescale.ovpworld.org/VybridSDHC
freescale.ovpworld.org/VybridSPI	freescale.ovpworld.org/VybridUART	freescale.ovpworld.org/VybridUSB
intel.ovpworld.org/82077AA	intel.ovpworld.org/82371EB	intel.ovpworld.org/8253
intel.ovpworld.org/8259A	intel.ovpworld.org/NorFlash48F4400	intel.ovpworld.org/PciIDE
intel.ovpworld.org/PciPM	intel.ovpworld.org/PciUSB	intel.ovpworld.org/Ps2Control
marvell.ovpworld.org/GT6412x	mips.ovpworld.org/16450C	

7.0 General Information on Peripheral Models

This document provides usage information for an Imperas OVP peripheral behavioral model.

The document is split into sections providing specific information for this peripheral, including any ports for connecting into a platform, registers etc. and configuration options and general information for peripheral modeling with Imperas OVP.

7.1 Background

Imperas OVP simulation technology enables very high performance simulation, debug and analysis of platforms containing multiple processors and peripheral models. The technology is designed to be extensible: you can create new models of processors, peripherals and other platform components using interfaces and libraries defined by OVP.

The peripheral models created using the OVP APIs run on the Peripheral Simulation Engine (PSE).

The model is typically written in C and compiled into an executable for the PSE processor architecture. The model is compiled for speed of execution and to protect IP. It is dynamically loaded by the simulator at run time.

8.0 Building peripherals easily with Imperas iGen

To aid with model creation, Imperas products include iGen, a model generation tool. iGen takes the laborious and error-prone task of constructing the various hardware model and software element files required for a typical model, and automates this process. iGen creates the needed C files. iGen also creates the C++ SystemC TLM2 interface files needed to run peripheral models in SystemC simulations.

iGen takes as input a simple script specification that includes device internals such as registers and memories, port information, component descriptors, and other elements. iGen then builds the C code model files and user editable templates. These include model frameworks with registers, function calls, memory map, and other items. It ensures that all component parts of the model are well-structured using best practices, and are consistent throughout the files, thus eliminating a common source of errors.

More information on iGen can be found: imperas.com/products.

Please contact Imperas to get access to the Imperas documents: Imperas_Model_Generator_Guide.pdf and Imperas_Peripheral_Generator_Guide.pdf.

9.0 Peripheral model internals

Each instance of a peripheral model runs on its own virtual machine with an address space large enough for the model. This processor (the PSE) and its memory are separate from any processors, memories and buses

in the platform being simulated; they exist only to execute the code of the peripheral model.

Interception of functions defined in the peripheral model allows the use of features of the host system in the implementation of the behavior of a peripheral. As an example, a real platform might contain a video display device. When simulating this system, it is generally more convenient not to simulate the complete video display device but to use a video package available on the host machine, such as SDL, and to use this to render to the host display. Also models of uarts, ethernet devices and USB components can make use of the host PC resources during simulation, to allow, for example, a simulation to browse the real internet, or the simulation to connect to a real USB device.

10.0 Parts of peripheral models

10.1 Configuring the Peripheral Instance with Parameters

A peripheral can include the behaviour of several configurations. These are controlled when the peripheral is instanced in the platform by setting parameters defined on the peripheral.

10.2 Net Ports

Peripherals may be connected to other peripherals or processors with signal wires (nets). These can be used to act as interrupt signals or used to control behavior between peripherals.
The wires are created in the platform as nets and this net is connected into the peripheral using a net port.

10.3 Bus master ports

A bus master port initiates (and controls the address of) a bus cycle. Bus cycles are generated by behavioral code within the peripheral model.

10.4 Bus slave ports

A peripheral can be defined as having several bus slave ports. The bus slave ports can be split into several address blocks. Each address block be either local memory or memory mapped registers. Both of these can have associated callback functions. A memory mapped register can also be defined as specific read/write access, whether it is volatile, and also whether it is associated with a reset pin and mask. A memory mapped register can also have specific bit fields defined.

10.5 Packetnets

A peripheral can be defined as being connected to packetnet ports. A packetnet is used to model packet based communication such as Ethernet, CAN bus or GSM. A packetnet is created in a platform, then connected to packetnet ports on model instances. A packetnet can have many connections, each able to send or receive packets. A packetnet is used as an efficient method of communication within OVP models.

For more information on modeling with packetnets, please see the peripheral modeling documentation: OVP_Peripheral_Modeling_Guide.pdf, OVPsim_and_CpuManager_User_Guide.pdf and the example: $IMPERAS_HOME/Examples/Models/Peripherals/packetnet.

11.0 More information (documentation) on peripheral models and modeling

More information on modeling and APIs can be found at: OVPworld.org/technology_apis.

Specifics on modeling peripherals can be found: OVP_Peripheral_Modeling_Guide.pdf.

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