OVP Guide to Using Processor Models

Model Specific Information for variant openCores

Imperas Software Limited
Imperas Buildings, North Weston
Thame, Oxfordshire, OX9 2HA, UK
docs@imperas.com

Author	Imperas Software Limited
Version	0.4
Filename	OVP_Model_Specific_Information_or1k_generic.pdf
Created	25 August 2015

Copyright Notice

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 reader's 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.

Table of Contents

1.0 Overview

This document provides the details of an OVP Fast Processor Model variant.

OVP Fast Processor Models are written in C and provide a C API for use in C based platforms. The models also provide a native interface for use in SystemC TLM2 platforms.

The models are written using the OVP VMI API that provides a Virtual Machine Interface 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.

Most 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 model families are validated using technology provided by the processor IP owners.

There is a companion document (OVP Guide to Using Processor Models) which explains the general concepts of OVP Fast Processor Models and their use. It is downloadable from the OVPworld website documentation pages.

1.1 Description

OR1K 32Bit processor model.

1.2 Licensing

Open Source Apache 2.0

1.3 Limitations

Core instruction set only. Does not support MMU & TLB.

2.0 Configuration

2.1 Location

The model source and object file is found in the VLNV tree at:
ovpworld.org/processor/or1k/1.0

2.2 GDB Path

The default GDB for this model is found at:
$IMPERAS_HOME/lib/$IMPERAS_ARCH/CrossCompiler/or32-elf/bin/or32-elf-gdb

2.3 Semi-Host Library

The default semi-host library file is found in the VLNV tree at :
ovpworld.org/semihosting/or1kNewlib/1.0

2.4 Processor Endian-ness

This is a BIG endian model.

2.5 QuantumLeap Support

This processor is qualified to run in a QuantumLeap enabled simulator.

2.6 Processor ELF Code

The ELF code supported by this model is: 0x8472

3.0 Other Variants in this Model

There are no different variants in this model.

4.0 Bus Ports

Table 1.

Type	Name	Bits	Description
master (initiator)	INSTRUCTION	32	Used to fetch code for execution
master (initiator)	DATA	32	Used to read & write data

5.0 Net Ports

Table 2.

Name	Type
intr0	input
intr1	input
intr2	input
intr3	input
reset	input

6.0 FIFO Ports

No FIFO Ports in this model.

7.0 Parameters

Table 3.

Name	Type	Description
fifos	Boolean	Turn on FIFO feature
verbose	Boolean	Turn on model messages

8.0 Execution Modes

Table 4.

Name	Code

SUPERVISOR	0
USER	1

9.0 Exceptions

Table 5.

Name	Code	Description
RST	256	Reset
BUS	512	Bus error
DPF	768	Data privilege
IPF	1024	Instruction privilege
TTI	1280	Tick timer
ILL	1792	Illegal instruction
EXI	2048	External interrupt
SYS	3072	System call

10.0 Hierarchy of the model

A CPU core may allow the user to configure it to instance many processors of a Symmetrical Multi Processor (SMP). A CPU core may also have sub elements within a processor, for example hardware threading blocks.

OVP processor models can be written to include SMP blocks and to have many levels of hierarchy.

Some OVP CPU models may have a fixed hierarchy, and some may be configured by settings in a configuration register. Please see the register definitions of this model.

This model documentation shows the settings and hierarchy of the default settings for this model variant.

10.1 Level 1:

This level in the model hierarchy has 2 commands.

This level in the model hierarchy has 3 register groups:

Table 6.

Group name	Registers
GPR	32
System	9
Integration_Support	1

This level in the model hierarchy has no children.

11.0 Model Commands

11.1 Level 1:

Table 7.

Name	Arguments
isync	specify instruction address range for synchronous execution
itrace	enable or disable instruction tracing

12.0 Registers

12.1 Level 1:

12.1.1 GPR

Table 8.

Name	Bits	Initial value (Hex)		Description
R0	32	0	r-	constant zero
R1	32	0	rw	stack pointer
R2	32	deadbeef	rw	frame pointer
R3	32	deadbeef	rw	
R4	32	deadbeef	rw	
R5	32	deadbeef	rw	
R6	32	deadbeef	rw	
R7	32	deadbeef	rw	
R8	32	deadbeef	rw	
R9	32	deadbeef	rw	
R10	32	deadbeef	rw	
R11	32	deadbeef	rw	
R12	32	deadbeef	rw	
R13	32	deadbeef	rw	
R14	32	deadbeef	rw	
R15	32	deadbeef	rw	
R16	32	deadbeef	rw	
R17	32	deadbeef	rw	
R18	32	deadbeef	rw	
R19	32	deadbeef	rw	
R20	32	deadbeef	rw	
R21	32	deadbeef	rw	
R22	32	deadbeef	rw	
R23	32	deadbeef	rw	

R24	32	deadbeef	rw	
R25	32	deadbeef	rw	
R26	32	deadbeef	rw	
R27	32	deadbeef	rw	
R28	32	deadbeef	rw	
R29	32	deadbeef	rw	
R30	32	deadbeef	rw	
R31	32	deadbeef	rw	

12.1.2 System

Table 9.

Name	**Bits**	**Initial value (Hex)**		**Description**
PC	32	0	rw	program counter
SR	32	8001	rw	status register
EPCR	32	deadbeef	rw	exception PC
EEAR	32	deadbeef	rw	exception effective address
ESR	32	deadbeef	rw	exception status register
PICMR	32	0	rw	PIC mask register
PICSR	32	0	rw	PIC status register
TTCR	32	0	rw	tick timer count register
TTMR	32	0	rw	tick timer mode register

12.1.3 Integration_Support

Table 10.

Name	**Bits**	**Initial value (Hex)**		**Description**
EXCPT	32	0	rw	current exception

#