OVP Guide to Using Processor Models

Model specific information for openCores_generic

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

Author	Imperas Software Limited
Version	20220722.0
Filename	OVP_Model_Specific_Information_or1k_generic.pdf
Created	27 July 2022
Status	OVP Standard Release

Model Release Status

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

Contents

Chapter 1

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

Feature	Implemented
Core instruction set	yes
MMU	no

TLB	no

Table 1.1: Implemented features

Chapter 2

Configuration

2.1 Location

This model's VLNV is ovpworld.org/processor/or1k/1.0.
The model source is usually at:
$IMPERAS_HOME/ImperasLib/source/ovpworld.org/processor/or1k/1.0
The model binary is usually at:
$IMPERAS_HOME/lib/$IMPERAS_ARCH/ImperasLib/ovpworld.org/processor/or1k/1.0

2.2 GDB Path

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

2.3 Semi-Host Library

The default semi-host library file is 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.

Chapter 3

All Variants in this model

This model has these variants

Variant	Description
generic	Single default variant

Table 3.1: All Variants in this model

Chapter 4

Bus Master Ports

This model has these bus master ports.

Name	min	max	Connect?	Description
INSTRUCTION	32	32	mandatory	Used to fetch code for execution
DATA	32	32	optional	Used to read & write data

Table 4.1: Bus Master Ports

Chapter 5

Bus Slave Ports

This model has no bus slave ports.

Chapter 6

Net Ports

This model has these net ports.

Name	Type	Connect?	Description
intr0	input	optional	
intr1	input	optional	
intr2	input	optional	
intr3	input	optional	
reset	input	optional	

Table 6.1: Net Ports

Chapter 7

FIFO Ports

This model has no FIFO ports.

Chapter 8

Formal Parameters

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

Table 8.1: Parameters

8.1 Parameter values

These are the current parameter values.

Name	Value
(Others)	
fifos	F
verbose	F
variant	generic

Table 8.2: Parameter values

Chapter 9

Execution Modes

Mode	Code
SUPERVISOR	0
USER	1

Table 9.1: Modes implemented in this processor

Chapter 10

Exceptions

Exception	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

Table 10.1: Exceptions implemented by this processor

Chapter 11

Hierarchy of the model

A CPU core may be configured 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.

11.1 Level 1

This level in the model hierarchy has 2 commands.
This level in the model hierarchy has 3 register groups:

Group name	Registers
GPR	32
System	9
Integration_Support	1

Table 11.1: Register groups

This level in the model hierarchy has no children.

Chapter 12

Model Commands

A Processor model can implement one or more **Model Commands** available to be invoked from the simulator command line, from the OP API or from the Imperas Multiprocessor Debugger.

12.1 Level 1

12.1.1 isync

specify instruction address range for synchronous execution

Argument	Type	Description
-addresshi	Uns64	end address of synchronous execution range
-addresslo	Uns64	start address of synchronous execution range

Table 12.1: isync command arguments

12.1.2 itrace

enable or disable instruction tracing

Argument	Type	Description
-after	Uns64	apply after this many instructions
-enable	Boolean	enable instruction tracing
-instructioncount	Boolean	include the instruction number in each trace
-memory	String	show memory accesses by this instruction. Argument can be any combination of X (execute), A (load or store access) and S (system)
-mode	Boolean	show processor mode changes
-off	Boolean	disable instruction tracing
-on	Boolean	enable instruction tracing
-processorname	Boolean	Include processor name in all trace lines
-registerchange	Boolean	show registers changed by this instruction
-registers	Boolean	show registers after each trace

Table 12.2: itrace command arguments

Chapter 13

Registers

13.1 Level 1

13.1.1 GPR

Registers at level:1, group:GPR

Name	Bits	Initial-Hex	RW	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	

Table 13.1: Registers at level 1, group:GPR

13.1.2 System

Registers at level:1, group:System

Name	Bits	Initial-Hex	RW	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

Table 13.2: Registers at level 1, group:System

13.1.3 Integration_Support

Registers at level:1, group:Integration_Support

Name	Bits	Initial-Hex	RW	Description
EXCPT	32	0	rw	current exception

Table 13.3: Registers at level 1, group:Integration_Support