



## OVP Guide to Using Processor Models

### Model Specific Information for variant MIPS64\_5Kc

#### Imperas Software Limited

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



Author	Imperas Software Limited
Version	0.5
Filename	OVP_Model_Specific_Information_mips64_5Kc.pdf
Created	27 February 2018
Status	OVP Standard Release

## **Copyright Notice**

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 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.

## **Model Release Status**

This model is released as part of OVP releases and is included in OVPworld packages. Please visit [OVPworld.org](http://OVPworld.org).

## Table of Contents

1 Overview.....	5
1.1 Description.....	5
1.2 Licensing.....	5
1.3 Limitations.....	5
1.4 Verification.....	5
1.5 Features.....	5
2 Configuration.....	5
2.1 Location.....	5
2.2 GDB Path.....	5
2.3 Semi-Host Library.....	5
2.4 Processor Endian-ness.....	6
2.5 QuantumLeap Support.....	6
2.6 Processor ELF Code.....	6
3 Other Variants in this Model.....	6
4 Bus Ports.....	6
5 Net Ports.....	6
6 FIFO Ports.....	6
7 Parameters.....	7
8 Execution Modes.....	10
9 Exceptions.....	11
10 Hierarchy of the model.....	12
10.1 Level 1: CPU.....	12
11 Model Commands.....	13
11.1 Level 1: CPU.....	13
11.1.1 isync.....	13
11.1.2 itrace.....	13
11.1.3 mipsCOP0.....	13
11.1.4 mipsCacheDisable.....	13
11.1.4.1 Argument description.....	13
11.1.5 mipsCacheEnable.....	13
11.1.6 mipsCacheRatio.....	14
11.1.7 mipsCacheReport.....	14
11.1.7.1 Argument description.....	14
11.1.8 mipsCacheReset.....	14
11.1.8.1 Argument description.....	14
11.1.9 mipsCacheTrace.....	14
11.1.10 mipsDebugFlags.....	14
11.1.11 mipsReadRegister.....	15
11.1.12 mipsReadTLBEntry.....	15
11.1.13 mipsTLBDump.....	15
11.1.13.1 Argument description.....	15
11.1.14 mipsTLBGetPhys.....	15

11.1.15 mipsWriteRegister.....	15
11.1.16 mipsWriteTLBEntry.....	15
12 Registers.....	16
12.1 Level 1: CPU.....	16
12.1.1 Core.....	16
12.1.2 DSP.....	17
12.1.3 COP0.....	17
12.1.4 Integration_support.....	18

## 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*

MIPS64 Configurable Processor Model

### *1.2 Licensing*

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

### *1.3 Limitations*

Cache model not implemented on mips64 variants

If this model is not part of your installation, then it is available for download from [www.OVPworld.org/MIPUser](http://www.OVPworld.org/MIPUser).

### *1.4 Verification*

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

### *1.5 Features*

MIPS64 Instruction set implemented

MMU Type: Standard TLB

Vectored interrupts implemented

## 2 Configuration

### *2.1 Location*

The model source and object file is found in the VLNV tree at:  
[mips.ovpworld.org/processor/mips64/1.0](http://mips.ovpworld.org/processor/mips64/1.0)

### *2.2 GDB Path*

The default GDB for this model is found at:

`$IMPERAS_HOME/lib/$IMPERAS_ARCH/gdb/mips-sde-elf-gdb`

### *2.3 Semi-Host Library*

The default semi-host library file is found in the VLNV tree at :  
mips.ovpworld.org/semihosting/mips64Newlib/1.0

### **2.4 Processor Endian-ness**

This model can be set to either endian-ness (normally by a pin, or the ELF code).

### **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: 0x8

## **3 Other Variants in this Model**

Table 1. All variants in this model

Variant
5Kf
5Kc
5KEf
5KEc

## **4 Bus Ports**

Table 2. Bus Ports

Type	Name	min	max
master (initiator)	INSTRUCTION	10	36
master (initiator)	DATA	10	36

## **5 Net Ports**

Table 3. Net Ports

Name	Type	Description
reset	input	Core reset
dint	input	Debug external interrupt
hwint0	input	External interrupt
hwint1	input	External interrupt
hwint2	input	External interrupt
hwint3	input	External interrupt
hwint4	input	External interrupt
hwint5	input	External interrupt
nmi	input	Non-maskable external interrupt

## **6 FIFO Ports**

No FIFO Ports in this model.

## 7 Parameters

Table 4. Parameters that can be set in the model, type: CPU

Name	Type	Description
mipsHexFile	String	Load a MIPS hex file (test-mode)
IMPERAS_MIPS_AVP_OPCODES	Boolean	Enable MIPS-specific magic Pass/Fail opcodes (specific for AVP test termination)
MIPS_TRACE	Boolean	Enable MIPS-format trace output
supervisorMode	Boolean	Override whether processor implements supervisor mode
busErrors	Boolean	Override bus error exception behavior. When true, accesses of memory not defined by platform will cause bus error exceptions
fixedMMU	Boolean	Override the MMU type to fixed mapping when true (sets Config.MT=3, Config.KU/K23=2 and Config1.MMUSizeM1=0)
removeDSP	Boolean	Override the DSP-present configuration when true (sets Config3.DSPP/DSP2P=0)
removeCMP	Boolean	Override the CMP-Present configuration when true (sets Config3.CMGCR and GCR_BASE to 0)
removeFP	Boolean	Override the FP-Present configuration when true (sets Config1.FP to 0)
isISA	Boolean	Enable to specify ISA model (reset address from ELF, all coprocessors enabled)
hiddenTLBentries	Boolean	Deprecated - Instead set config1MMUSizeM1 to maximum value to improve performance
ITCNumEntries	Uns32	Specify number of ITC cells present (MT cores only)
ITCNumFIFO	Uns32	Specify number of ITC FIFO cells in reference ITC implementation (MT cores only)
MTFPU	Uns32	Enable multi-threaded FPU (1:old mttc1 behavior, 2:new mttc1 behavior)
supportDenormals	Boolean	Enable to specify that the FPU supports denormal operands and results
VPE0MaxTC	Uns32	Specifies the maximum TCs initially on VPE0
segBits	Uns32	Override the number of address bits implemented for 64 bit segments (MIPS64 Only)
mpuRegions	Uns32	Number of regions for memory protection unit
mpuType	Uns32	Type of MPU implementation
mpuEnable	Boolean	Enable MPU2 segment control at reset
mpuSegment0	Uns32	Attributes for segment 0 in MPU2 SegmentControl_0 register
mpuSegment1	Uns32	Attributes for segment 1 in MPU2 SegmentControl_0 register

mpuSegment2	Uns32	Attributes for segment 2 in MPU2 SegmentControl_0 register
mpuSegment3	Uns32	Attributes for segment 3 in MPU2 SegmentControl_0 register
mpuSegment4	Uns32	Attributes for segment 4 in MPU2 SegmentControl_1 register
mpuSegment5	Uns32	Attributes for segment 5 in MPU2 SegmentControl_1 register
mpuSegment6	Uns32	Attributes for segment 6 in MPU2 SegmentControl_1 register
mpuSegment7	Uns32	Attributes for segment 7 in MPU2 SegmentControl_1 register
mpuSegment8	Uns32	Attributes for segment 8 in MPU2 SegmentControl_2 register
mpuSegment9	Uns32	Attributes for segment 9 in MPU2 SegmentControl_2 register
mpuSegment10	Uns32	Attributes for segment 10 in MPU2 SegmentControl_2 register
mpuSegment11	Uns32	Attributes for segment 11 in MPU2 SegmentControl_2 register
mpuSegment12	Uns32	Attributes for segment 12 in MPU2 SegmentControl_3 register
mpuSegment13	Uns32	Attributes for segment 13 in MPU2 SegmentControl_3 register
mpuSegment14	Uns32	Attributes for segment 14 in MPU2 SegmentControl_3 register
mpuSegment15	Uns32	Attributes for segment 15 in MPU2 SegmentControl_3 register
mvpcnf0vpe	Uns32	Override MVPConf0.PVPE
mvpcnf0tc	Uns32	Override MVPConf0.PTC
mvpcnf0pcp	Boolean	Override MVPConf0.PCP
mvpcnf0tcp	Boolean	Override MVPConf0.TCP
hasFDC	Uns32	Specify the size of Fast Debug Channel register block
statusFR	Boolean	Override power on value in Status.FR (Floating point register mode)
configDSP	Boolean	Override Config.DSP (data scratchpad RAM present)
configISP	Boolean	Override Config.ISP (instruction scratchpad RAM present)
configK0	Uns32	Override power on value of Config.K0 (set Kseg0 cacheability)
configKU	Uns32	Override power on value of Config.KU (set Useg cacheability)
configK23	Uns32	Override power on value of Config.K23 (set Kseg23 cacheability)



configMDU	Boolean	Override Config.MDU (iterative multiply/divide unit)
configMM	Boolean	Override Config.MM (merging mode for write)
configMT	Uns32	Override Config.MT
configSB	Boolean	Override Config.SB (simple bus transfers only)
MIPS16eASE	Boolean	Override Config1.CA (enables the MIPS16e ASE)
config1EP	Boolean	Override Config1.EP (EJTag present)
config1MMUSizeM1	Uns32	Override Config1.MMUSizeM1 (number of MMU entries-1)
config1WR	Boolean	Override Config1.WR (watchpoint registers present)
config1FP	Boolean	Override Config1.FP (FPU present)
config3BI	Boolean	Override Config3.BI
config3BP	Boolean	Override Config3.BP
config3CDMM	Boolean	Override Config3.CDMM
config3CTXTC	Boolean	Override Config3.CTXTC
config3DSPP	Boolean	Override Config3.DSPP
config3DSP2P	Boolean	Override Config3.DSP2P
config3IPLW	Uns32	Override Config3.IPLW
config3ISA	Uns32	Override Config3.ISA
config3ISAOnExc	Boolean	Override Config3.ISAOnExc
config3ITL	Boolean	Override Config3.ITL
config3LPA	Boolean	Override Config3.LPA
config3MCU	Boolean	Override Config3.MCU
config3MMAR	Uns32	Override Config3.MMAR
config3RXI	Boolean	Override Config3.RXI
config3SC	Boolean	Override Config3.SC
config3ULRI	Boolean	Override Config3.ULRI
externalinterrupt	Boolean	Override Config3.VEIC (enables the use of an external interrupt controller)
vectoredinterrupt	Boolean	Override Config3.VInt (enables vectored interrupts)
config3VZ	Boolean	Override Config3.VZ
config4AE	Boolean	Override Config4.AE
config4IE	Uns32	Override Config4.IE
config4MMUConfig	Uns32	Override Config4.MMUConfig field (interpretation depends on MMUExtDef value)
config4MMUExtDef	Uns32	Override Config4.MMUExtDef
config4VTLBSizeExt	Uns32	Override Config4.VTLBSizeExt
config5EVA	Boolean	Override Config5.EVA
config5NFExists	Boolean	Override Config5.NFExists
config5MSAEn	Boolean	Override Config5.MSAEn

config6FTLBE	Boolean	Override power on value of Config6.FTLBE
config7DCIDX_MODE	Uns32	Override Config7.DCIDX_MODE
config7WII	Boolean	Override Config7.WII (wait IE/IXMT ignore)
fcsrABS2008	Boolean	Override FCSR.ABS2008 (ABS/NEG compliant with IEEE 754-2008)
fcsrNaN2008	Boolean	Override FCSR.NaN2008 (QNaN/SNaN encodings match IEEE 754-2008 recommendation)
firPS	Boolean	Override FIR.PS (PS floating point type implemented)
firHas2008	Boolean	Override FIR.Has2008 (one or more IEEE 754-2008 features present)
intctlIPFDC	Uns32	Override IntCtl.IPFDC
intctlIPTI	Uns32	Override IntCtl.IPTI
pridRevision	Uns32	Override PRId.Revision
srscctlHSS	Uns32	Override SRSCtl.HSS (number of shadow register sets)
ExceptionBase	Uns32	Specify the BEV Exception Base address. (use GCR_Cx_RESET_BASE on CMP processors)
UseExceptionBase	Boolean	Set to one to use ExceptionBase[29:12] as the corresponding BEV address bits
EIC_OPTION	Uns32	Override the external interrupt controller EIC_OPTION
ISPRAM_SIZE	Uns32	Encoded size of the ISPRAM region (log2(<ISPRAM size in bytes>) - 11)
ISPRAM_BASE	Uns64	Starting physical address of the ISPRAM region
ISPRAM_ENABLE	Boolean	Set the enable bit of the ISPRAM region's tag (used to enable the ISPRAM region prior to reset)
ISPRAM_FILE	String	Load a MIPS hex file into the ISPRAM region prior to reset
DSPRAM_SIZE	Uns32	Encoded size of the DSPRAM region (log2(<DSPRAM size in bytes>) - 11)
DSPRAM_BASE	Uns64	Starting physical address of the DSPRAM region
DSPRAM_ENABLE	Boolean	Set the enable bit of the DSPRAM region's tag (used to enable the DSPRAM region prior to reset)

## 8 Execution Modes

Table 5. CPU modes implemented in the model, type: CPU

Name	Code
KERNEL	0

DEBUG	1
SUPERVISOR	2
USER	3

## 9 Exceptions

Table 6. Exceptions handled by the model, type: CPU

Name	Code
Int	0
Mod	1
TLBL	2
TLBS	3
AdEL	4
AdES	5
IBE	6
DBE	7
Sys	8
Bp	9
RI	10
CpU	11
Ov	12
Tr	13
FPE	15
Impl1	16
Impl2	17
C2E	18
TLBRI	19
TLBXI	20
MDMX	22
WATCH	23
MCheck	24
Thread	25
DSPDis	26
Prot	29
CacheErr	30

## 10 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: CPU

This level in the model hierarchy has 16 commands.

This level in the model hierarchy has 4 register groups:

Table 7. Register groups

Group name	Registers
Core	33
DSP	9
COP0	32
Integration_support	1

This level in the model hierarchy has no children.

## 11 Model Commands

### 11.1 Level 1: CPU

#### 11.1.1 *isync*

specify instruction address range for synchronous execution

Table 8. *isync* command arguments

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

#### 11.1.2 *itrace*

enable or disable instruction tracing

Table 9. *itrace* command arguments

Argument	Type	Description
-after	Uns64	apply after this many instructions
-enable	Boolean	enable instruction tracing
-instructioncount	Boolean	include the instruction number in each trace
-off	Boolean	disable instruction tracing
-on	Boolean	enable instruction tracing
-registerchange	Boolean	show registers changed by this instruction
-registers	Boolean	show registers after each trace

#### 11.1.3 *mipsCOP0*

query a COP0 register value using <register> <select>

Table 10. *mipsCOP0* command arguments

Argument	Type	Description
-register	Int32	specify the COP0 register number
-select	Int32	specify the COP0 register select

#### 11.1.4 *mipsCacheDisable*

##### 11.1.4.1 *Argument description*

Disables tag or full cache model

#### 11.1.5 *mipsCacheEnable*

enable tag or full cache model

Table 11. *mipsCacheEnable* command arguments

Argument	Type	Description
----------	------	-------------

-debug	Int32	set cache model debug flags
-full	Boolean	enable full cache model
-tag	Boolean	enable cache tag line only model

### ***11.1.6 mipsCacheRatio***

Report current hit ratio for selected cache

Table 12. mipsCacheRatio command arguments

<b>Argument</b>	<b>Type</b>	<b>Description</b>
-dcache	Boolean	report hit ratio for dcache
-icache	Boolean	report hit ratio for icache

### ***11.1.7 mipsCacheReport***

#### ***11.1.7.1 Argument description***

Report current cache statistics

### ***11.1.8 mipsCacheReset***

#### ***11.1.8.1 Argument description***

reset the cache model

### ***11.1.9 mipsCacheTrace***

Control the tracing of cache accesses

Table 13. mipsCacheTrace command arguments

<b>Argument</b>	<b>Type</b>	<b>Description</b>
-noartifact	Boolean	filter artifact accesses
-nocached	Boolean	filter cached accesses
-nodcache	Boolean	filter dcache accesses
-noicache	Boolean	filter icache accesses
-notrue	Boolean	filter true accesses
-nouncached	Boolean	filter uncached accesses
-off	Boolean	turn off the cache tracing
-on	Boolean	turn on the cache tracing

### ***11.1.10 mipsDebugFlags***

Set the processor model debug flags to <value>

Table 14. mipsDebugFlags command arguments

<b>Argument</b>	<b>Type</b>	<b>Description</b>
-----------------	-------------	--------------------

-value	Int32	specify model debug flags
--------	-------	---------------------------

### ***11.1.11 mipsReadRegister***

Read a processor register using <resource> <offset>

Table 15. mipsReadRegister command arguments

Argument	Type	Description
-offset	Int32	the processor register offset
-resource	Int32	the processor register resource number

### ***11.1.12 mipsReadTLBEntry***

read a TLB entry specified by the index

Table 16. mipsReadTLBEntry command arguments

Argument	Type	Description
-index	Uns64	select the TLB entry

### ***11.1.13 mipsTLBDump***

#### ***11.1.13.1 Argument description***

Dumps the current contents of the TLB

### ***11.1.14 mipsTLBGetPhys***

Reports the entry(s) in the TLB that match the given virtual address and ASID

Table 17. mipsTLBGetPhys command arguments

Argument	Type	Description
-asid	Uns64	ASID
-va	Uns64	virtual address

### ***11.1.15 mipsWriteRegister***

Write to a processor register using <resource> <offset> <value>

Table 18. mipsWriteRegister command arguments

Argument	Type	Description
-offset	Int32	the register offset number
-resource	Int32	the register resource number
-value	Uns64	the register value to be written

### ***11.1.16 mipsWriteTLBEntry***

Writes values to a TLB entry using the index, lo0, lo1, hi0 and mask fields

Table 19. mipsWriteTLBEntry command arguments

Argument	Type	Description
-hi0	Uns64	the TLB entry high address
-index	Uns64	the TLB entry index
-lo0	Uns64	the TLB entry low address 0
-lo1	Uns64	the TLB entry low address 1
-mask	Uns64	the TLB entry mask

## 12 Registers

### 12.1 Level 1: CPU

#### 12.1.1 Core

Table 20. Registers at level 1, type: CPU, register group: 'Core'

Name	Bits	Initial value (Hex)		Description
zero	64	0	r-	constant zero
at	64	0	rw	
v0	64	0	rw	
v1	64	0	rw	
a0	64	0	rw	
a1	64	0	rw	
a2	64	0	rw	
a3	64	0	rw	
t0	64	0	rw	
t1	64	0	rw	
t2	64	0	rw	
t3	64	0	rw	
t4	64	0	rw	
t5	64	0	rw	
t6	64	0	rw	
t7	64	0	rw	
s0	64	0	rw	
s1	64	0	rw	
s2	64	0	rw	
s3	64	0	rw	
s4	64	0	rw	
s5	64	0	rw	
s6	64	0	rw	
s7	64	0	rw	
t8	64	0	rw	



t9	64	0	rw	
k0	64	0	rw	
k1	64	0	rw	
gp	64	0	rw	
sp	64	0	rw	stack pointer
s8	64	0	rw	frame pointer
ra	64	0	rw	
pc	64	ffffffbfc00000	rw	program counter

### 12.1.2 DSP

Table 21. Registers at level 1, type: CPU, register group: 'DSP'

Name	Bits	Initial value (Hex)		Description
lo	64	0	rw	
hi	64	0	rw	
lo1	64	0	rw	
hi1	64	0	rw	
lo2	64	0	rw	
hi2	64	0	rw	
lo3	64	0	rw	
hi3	64	0	rw	
dspctl	64	0	rw	DSP control

### 12.1.3 COP0

Table 22. Registers at level 1, type: CPU, register group: 'COP0'

Name	Bits	Initial value (Hex)		Description
sr	64	400004	rw	CP0 register 12/0 (status)
bad	64	0	rw	CP0 register 8/0 (badvaaddr)
cause	64	0	rw	CP0 register 13/0 (cause)
index	64	0	rw	CP0 register 0/0
random	64	0	rw	CP0 register 1/0
entrylo0	64	0	rw	CP0 register 2/0
entrylo1	64	0	rw	CP0 register 3/0
context	64	0	rw	CP0 register 4/0
pagemask	64	0	rw	CP0 register 5/0
wired	64	0	rw	CP0 register 6/0
hwrena	64	0	rw	CP0 register 7/0
badvaddr	64	0	rw	CP0 register 8/0
count	64	0	rw	CP0 register 9/0
entryhi	64	0	rw	CP0 register 10/0
compare	64	0	rw	CP0 register 11/0

status	64	400004	rw	CP0 register 12/0
intctl	64	fc000000	rw	CP0 register 12/1
srsctl	64	0	rw	CP0 register 12/2
srsmap	64	0	rw	CP0 register 12/3
epc	64	0	rw	CP0 register 14/0
prid	64	18100	rw	CP0 register 15/0
ebase	64	fffffff800000000	rw	CP0 register 15/1
config	64	b600c483	rw	CP0 register 16/0
config1	64	dee37182	rw	CP0 register 16/1
config2	64	80000000	rw	CP0 register 16/2
config3	64	20	rw	CP0 register 16/3
lladdr	64	0	rw	CP0 register 17/0
xcontext	64	0	rw	CP0 register 20/0
debug	64	2010000	rw	CP0 register 23/0
depc	64	0	rw	CP0 register 24/0
errorepc	64	0	rw	CP0 register 30/0
desave	64	0	rw	CP0 register 31/0

#### *12.1.4 Integration\_support*

Table 23. Registers at level 1, type: CPU, register group: 'Integration\_support'

Name	Bits	Initial value (Hex)		Description
stop	32	0	rw	write with non-zero to stop processor

#