OVP Guide to Using Processor Models

Model specific information for
RISC-V_RV32IM

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

<table>
<thead>
<tr>
<th>Author</th>
<th>Imperas Software Limited</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>20220722.0</td>
</tr>
<tr>
<td>Filename</td>
<td>OVP_Model_Specific_Information_riscv_RV32IM.pdf</td>
</tr>
<tr>
<td>Created</td>
<td>27 July 2022</td>
</tr>
<tr>
<td>Status</td>
<td>OVP Standard Release</td>
</tr>
</tbody>
</table>
Copyright Notice
Copyright (c) 2022 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.
# Contents

1 Overview ................................................................. 1
   1.1 Description ....................................................... 1
   1.2 Licensing ......................................................... 1
   1.3 Extensions ......................................................... 2
      1.3.1 Extensions Enabled by Default ............................. 2
      1.3.2 Enabling Other Extensions ................................. 2
      1.3.3 Disabling Extensions ...................................... 3
   1.4 General Features .................................................. 3
      1.4.1 mtvec CSR .................................................... 3
      1.4.2 stvec CSR .................................................... 4
      1.4.3 Reset ......................................................... 4
      1.4.4 NMI ........................................................... 4
      1.4.5 WFI .......................................................... 4
      1.4.6 cycle CSR .................................................... 4
      1.4.7 instret CSR .................................................. 4
      1.4.8 hpmcounter CSR ............................................. 5
      1.4.9 time CSR ..................................................... 5
      1.4.10 mcycle CSR ................................................ 5
      1.4.11 minstret CSR ............................................... 5
      1.4.12 mhpmcounter CSR ......................................... 5
      1.4.13 Virtual Memory ............................................. 5
      1.4.14 Unaligned Accesses ....................................... 6
      1.4.15 PMP ........................................................ 6
   1.5 Privileged Architecture ........................................... 6
      1.5.1 Legacy Version 1.10 ....................................... 6
      1.5.2 Version 20190608 ......................................... 6
      1.5.3 Version 20211203 .......................................... 7
      1.5.4 Version 1.12 ............................................... 7
      1.5.5 Version master .............................................. 7
   1.6 Unprivileged Architecture ....................................... 7
      1.6.1 Legacy Version 2.2 ....................................... 7
      1.6.2 Version 20191213 ......................................... 7
   1.7 Other Extensions .................................................. 8
      1.7.1 Zmmul ........................................................ 8
      1.7.2 Zicsr ........................................................ 8
      1.7.3 Zifencei ...................................................... 8
      1.7.4 Zibcom ....................................................... 8
8.1.6 Parameter smepmp_version ........................................ 27
8.1.7 Parameter debug_mode ........................................... 27
8.2 Parameter values .................................................. 27

9 Execution Modes ..................................................... 31

10 Exceptions .......................................................... 32

11 Hierarchy of the model .............................................. 33
  11.1 Level 1: Hart .................................................. 33

12 Model Commands .................................................... 34
  12.1 Level 1: Hart .................................................. 34
    12.1.1 debugflags ............................................. 34
    12.1.2 dumpTLB .................................................. 34
      12.1.2.1 Argument description ................................ 34
    12.1.3 getCSRIndex ............................................. 34
    12.1.4 isync .................................................... 34
    12.1.5 itrace ................................................... 35
    12.1.6 listCSRs .................................................. 35
      12.1.6.1 Argument description ................................ 35

13 Registers .......................................................... 36
  13.1 Level 1: Hart .................................................. 36
    13.1.1 Core .................................................... 36
    13.1.2 User-Control_and_Status ................................ 37
    13.1.3 Supervisor-Control_and_Status ......................... 38
    13.1.4 Machine-Control_and_Status ............................ 38
    13.1.5 Integration_support ................................... 42
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

RISC-V RV32IM 32-bit processor model

1.2 Licensing

This Model is released under the Open Source Apache 2.0
1.3 Extensions

1.3.1 Extensions Enabled by Default

The model has the following architectural extensions enabled, and the corresponding bits in the misa CSR Extensions field will be set upon reset:

- misa bit 8: RV32I/RV64I/RV128I base integer instruction set
- misa bit 12: extension M (integer multiply/divide instructions)
- misa bit 18: extension S (Supervisor mode)
- misa bit 20: extension U (User mode)

To specify features that can be dynamically enabled or disabled by writes to the misa register in addition to those listed above, use parameter “add_Extensions_mask”. This is a string parameter containing the feature letters to add; for example, value “DV” indicates that double-precision floating point and the Vector Extension can be enabled or disabled by writes to the misa register, if supported on this variant. Parameter “sub_Extensions_mask” can be used to disable dynamic update of features in the same way.

Legacy parameter “misa_Extensions_mask” can also be used. This Uns32-valued parameter specifies all writable bits in the misa Extensions field, replacing any permitted bits defined in the base variant.

Note that any features that are indicated as present in the misa mask but absent in the misa will be ignored. See the next section.

1.3.2 Enabling Other Extensions

The following extensions are supported by the model, but not enabled by default in this variant:

- misa bit 0: extension A (atomic instructions)
- misa bit 1: extension B (bit manipulation extension)
- misa bit 2: extension C (compressed instructions)
- misa bit 3: extension D (double-precision floating point)
- misa bit 4: RV32E base integer instruction set (embedded)
- misa bit 5: extension F (single-precision floating point)
- misa bit 7: extension H (hypervisor)
- misa bit 10: extension K (cryptographic)
- misa bit 13: extension N (user-level interrupts)
- misa bit 15: extension P (DSP instructions)
- misa bit 21: extension V (vector extension)
- misa bit 23: extension X (non-standard extensions present)
To add features from this list to the visible set in the misa register, use parameter “add_Extensions”. This is a string containing identification letters of features to enable; for example, value “DV” indicates that double-precision floating point and the Vector Extension should be enabled, if they are currently absent and are available on this variant.

Legacy parameter “misa_Extensions” can also be used. This Uns32-valued parameter specifies the reset value for the misa CSR Extensions field, replacing any permitted bits defined in the base variant.

To add features from this list to the implicitly-enabled set (not visible in the misa register), use parameter “add_implicit_Extensions”. This is a string parameter in the same format as the “add_Extensions” parameter described above.

### 1.3.3 Disabling Extensions

The following extensions are enabled by default in the model and can be disabled:

- misa bit 12: extension M (integer multiply/divide instructions)
- misa bit 18: extension S (Supervisor mode)
- misa bit 20: extension U (User mode)

To disable features that are enabled by default, use parameter “sub_Extensions”. This is a string containing identification letters of features to disable; for example, value “DF” indicates that double-precision and single-precision floating point extensions should be disabled, if they are enabled by default on this variant.

To remove features from this list from the implicitly-enabled set (not visible in the misa register), use parameter “sub_implicit_Extensions”. This is a string parameter in the same format as the “sub_Extensions” parameter described above.

### 1.4 General Features

#### 1.4.1 mtvec CSR

On this variant, the Machine trap-vector base-address register (mtvec) is writable. It can instead be configured as read-only using parameter “mtvec_is_ro”.

Values written to “mtvec” are masked using the value 0xffffffff. A different mask of writable bits may be specified using parameter “mtvec_mask” if required. In addition, when Vectored interrupt mode is enabled, parameter “tvec_align” may be used to specify additional hardware-enforced base address alignment. In this variant, “tvec_align” defaults to 0, implying no alignment constraint.

If parameter “mtvec_sext” is True, values written to “mtvec” are sign-extended from the most-significant writable bit. In this variant, “mtvec_sext” is False, indicating that “mtvec” is not sign-extended.

The initial value of “mtvec” is 0x0. A different value may be specified using parameter “mtvec” if required.
1.4.2 stvec CSR

Values written to “stvec” are masked using the value 0xfffffffd. A different mask of writable bits may be specified using parameter “stvec_mask” if required. In addition, when Vectored interrupt mode is enabled, parameter “tvec_align” may be used to specify additional hardware-enforced base address alignment. In this variant, “tvec_align” defaults to 0, implying no alignment constraint.

If parameter “stvec_sext” is True, values written to “stvec” are sign-extended from the most-significant writable bit. In this variant, “stvec_sext” is False, indicating that “stvec” is not sign-extended.

1.4.3 Reset

On reset, the model will restart at address 0x0. A different reset address may be specified using parameter “reset_address” or applied using optional input port “reset_addr” if required.

1.4.4 NMI

On an NMI, the model will restart at address 0x0; a different NMI address may be specified using parameter “nmi_address” or applied using optional input port “nmi_addr” if required. The cause reported on an NMI is 0x0 by default; a different cause may be specified using parameter “ecode_nmi” or applied using optional input port “nmi_cause” if required.

If parameter “rnmi_version” is not “none”, resumable NMIs are supported, managed by additional CSRs “mnscratch”, “mnepc”, “mncause” and “mnstatus”, following the indicated version of the Resumable NMI extension proposal. In this variant, “rnmi_version” is “none”.

The NMI input is level-sensitive. To instead specify that the NMI input is latched on the rising edge of the NMI signal, set parameter “nmi_is_latched” to True.

1.4.5 WFI

WFI will halt the processor until an interrupt occurs. It can instead be configured as a NOP using parameter “wfi_is_nop”. WFI timeout wait is implemented with a time limit of 0 (i.e. WFI causes an Illegal Instruction trap in Supervisor mode when mstatus.TW=1).

1.4.6 cycle CSR

The “cycle” CSR is implemented in this variant. Set parameter “cycle_undefined” to True to instead specify that “cycle” is unimplemented and accesses should cause Illegal Instruction traps.

1.4.7 instret CSR

The “instret” CSR is implemented in this variant. Set parameter “instret_undefined” to True to instead specify that “instret” is unimplemented and accesses should cause Illegal Instruction traps.
1.4.8 hpmcounter CSR

The “hpmcounter” CSRs are implemented in this variant. Set parameter “hpmcounter_undefined” to True to instead specify that “hpmcounter” CSRs are unimplemented and accesses should cause Illegal Instruction traps.

1.4.9 time CSR

The “time” CSR is implemented in this variant. Set parameter “time_undefined” to True to instead specify that “time” is unimplemented and reads of it should cause Illegal Instruction traps. Usually, the value of the “time” CSR should be provided by the platform - see notes below about the artifact “CSR” bus for information about how this is done.

1.4.10 mcycle CSR

The “mcycle” CSR is implemented in this variant. Set parameter “mcycle_undefined” to True to instead specify that “mcycle” is unimplemented and accesses should cause Illegal Instruction traps.

1.4.11 minstret CSR

The “minstret” CSR is implemented in this variant. Set parameter “minstret_undefined” to True to instead specify that “minstret” is unimplemented and accesses should cause Illegal Instruction traps.

1.4.12 mhpmcounter CSR

The “mhpmcounter” CSRs are implemented in this variant. Set parameter “mhpmcounter_undefined” to True to instead specify that “mhpmcounter” CSRs are unimplemented and accesses should cause Illegal Instruction traps.

1.4.13 Virtual Memory

This variant supports address translation modes 0 (bare) and 1 (Sv32). Use parameter “Sv_modes” to specify a bit mask of different implemented modes if required; for example, setting “Sv_modes” to (1<<0)+(1<<8) indicates that mode 0 (bare) and mode 8 (Sv39) are implemented. These indices correspond to writable values in the satp.MODE CSR field.

A 9-bit ASID is implemented. Use parameter “ASID_bits” to specify a different implemented ASID size if required.

TLB behavior is controlled by parameter “ASIDCacheSize”. If this parameter is 0, then an unlimited number of TLB entries will be maintained concurrently. If this parameter is non-zero, then only TLB entries for up to “ASIDCacheSize” different ASIDs will be maintained concurrently initially; as new ASIDs are used, TLB entries for less-recently used ASIDs are deleted, which improves model performance in some cases. If the model detects that the TLB entry cache is too small
(entry ejections are very frequent), it will increase the cache size automatically. In this variant, “ASIDCacheSize” is 8.

1.4.14 Unaligned Accesses

Unaligned memory accesses are not supported by this variant. Set parameter “unaligned” to “T” to enable such accesses.

Address misaligned exceptions are higher priority than page fault or access fault exceptions on this variant. Set parameter “unaligned_low_pri” to “T” to specify that they are lower priority instead.

1.4.15 PMP

16 PMP entries are implemented by this variant. Use parameter “PMP_registers” to specify a different number of PMP entries; set the parameter to 0 to disable the PMP unit. The PMP grain size (G) is 0, meaning that PMP regions as small as 4 bytes are implemented. Use parameter “PMP_grain” to specify a different grain size if required. Unaligned PMP accesses are not decomposed into separate aligned accesses; use parameter “PMP_decompose” to modify this behavior if required. Parameters to change the write masks for the PMP CSRs are not enabled; use parameter “PMP_maskparams” to modify this behavior if required. Parameters to change the reset values for the PMP CSRs are not enabled; use parameter “PMP_initialparams” to modify this behavior if required.

Accesses to unimplemented PMP registers are write-ignored and read as zero on this variant. Set parameter “PMP_undefined” to True to indicate that such accesses should cause Illegal Instruction exceptions instead.

1.5 Privileged Architecture

This variant implements the Privileged Architecture with version specified in the References section of this document. Note that parameter “priv_version” can be used to select the required architecture version; see the following sections for detailed information about differences between each supported version.

1.5.1 Legacy Version 1.10

1.10 version of May 7 2017.

1.5.2 Version 20190608

Stable 1.11 version of June 8 2019, with these changes compared to version 1.10:

- mcountinhibit CSR defined;
- pages are never executable in Supervisor mode if page table entry U bit is 1;
- mstatus.TW is writable if any lower-level privilege mode is implemented (previously, it was just if Supervisor mode was implemented);

1.5.3 Version 20211203

1.12 draft version of December 3 2021, with these changes compared to version 20190608:
- mstathush, mseccfg, mseccfgh, menvcfg, menvcfgh, senvcfg, henvcfg, henvcfgh and mconfigptr CSRs defined;
- xret instructions clear mstatus.MPRV when leaving Machine mode if new mode is less privileged than M-mode;
- maximum number of PMP registers increased to 64;
- data endian is now configurable.

1.5.4 Version 1.12

Official 1.12 version, identical to 20211203.

1.5.5 Version master

Unstable master version, currently identical to 1.12.

1.6 Unprivileged Architecture

This variant implements the Unprivileged Architecture with version specified in the References section of this document. Note that parameter “user_version” can be used to select the required architecture version; see the following sections for detailed information about differences between each supported version.

1.6.1 Legacy Version 2.2

2.2 version of May 7 2017.

1.6.2 Version 20191213

Stable 20191213-Base-Ratified version of December 13 2019, with these changes compared to version 2.2:
- floating point fmin/fmax instruction behavior modified to comply with IEEE 754-201x.
- numerous other optional behaviors can be separately enabled using Z-prefixed parameters.
1.7 Other Extensions

Other extensions that can be configured are described in this section.

1.7.1 Zmmul

Parameter “Zmmul” is 0 on this variant, meaning that all multiply and divide instructions are implemented. if “Zmmul” is set to 1 then multiply instructions are implemented but divide and remainder instructions are not implemented.

1.7.2 Zicsr

Parameter “Zicsr” is 1 on this variant, meaning that standard CSRs and CSR access instructions are implemented. if “Zicsr” is set to 0 then standard CSRs and CSR access instructions are not implemented and an alternative scheme must be provided as a processor extension.

1.7.3 Zifencei

Parameter “Zifencei” is 1 on this variant, meaning that the fence.i instruction is implemented (but treated as a NOP by the model). if “Zifencei” is set to 0 then the fence.i instruction is not implemented.

1.7.4 Zicbom

Parameter “Zicbom” is 0 on this variant, meaning that code block management instructions are undefined. If “Zicbom” is set to 1 then code block management instructions cbo.clean, cbo.flush and cbo.inval are defined.

If Zicbom is present, the cache block size is given by parameter “cmomp_bytes”. The instructions may cause traps if used illegally but otherwise are NOPs in this model.

1.7.5 Zicbop

Parameter “Zicbop” is 0 on this variant, meaning that prefetch instructions are undefined. If “Zicbop” is set to 1 then prefetch instructions prefetch.i, prefetch.r and prefetch.w are defined (but behave as NOPs in this model).

1.7.6 Zicboz

Parameter “Zicboz” is 0 on this variant, meaning that the cbo.zero instruction is undefined. If “Zicboz” is set to 1 then the cbo.zero instruction is defined.

If Zicboz is present, the cache block size is given by parameter “cmoz_bytes”.

Copyright (c) 2022 Imperas Software Limited www.ovpworld.org
OVP License. Release 20220722.0
1.7.7 Smstateen

Parameter “Smstateen” is 0 on this variant, meaning that state enable CSRs are undefined. If “Smstateen” is set to 1 then state enable CSRs are defined.

Within the state enable CSRs, only bit 1 (for Zfinx), bit 57 (for xcontext CSR access), bit 62 (for xenvcfg CSR access) and bit 63 (for lower-level state enable CSR access) are currently implemented.

1.8 CLIC

The model can be configured to implement a Core Local Interrupt Controller (CLIC) using parameter “CLICLEVELS”; when non-zero, the CLIC is present with the specified number of interrupt levels (2-256), as described in the RISC-V Core-Local Interrupt Controller specification, and further parameters are made available to configure other aspects of the CLIC. “CLICLEVELS” is zero in this variant, indicating that a CLIC is not implemented.

1.9 Interrupts

The “reset” port is an active-high reset input. The processor is halted when “reset” goes high and resumes execution from the reset address specified using the “reset_address” parameter or “reset_addr” port when the signal goes low. The “mcause” register is cleared to zero.

The “nmi” port is an active-high NMI input. The processor resumes execution from the address specified using the “nmi_address” parameter or “nmi_addr” port when the NMI signal goes high. The “mcause” register is cleared to zero.

All other interrupt ports are active high. For each implemented privileged execution level, there are by default input ports for software interrupt, timer interrupt and external interrupt; for example, for Machine mode, these are called “MSWInterrupt”, “MTimerInterrupt” and “MExternalInterrupt”, respectively. When the N extension is implemented, ports are also present for User mode. Parameter “unimp_int_mask” allows the default behavior to be changed to exclude certain interrupt ports. The parameter value is a mask in the same format as the “mip” CSR; any interrupt corresponding to a non-zero bit in this mask will be removed from the processor and read as zero in “mip”, “mie” and “mideleg” CSRs (and Supervisor and User mode equivalents if implemented).

Parameter “external_int_id” can be used to enable extra interrupt ID input ports on each hart. If the parameter is True then when an external interrupt is applied the value on the ID port is sampled and used to fill the Exception Code field in the “mcause” CSR (or the equivalent CSR for other execution levels). For Machine mode, the extra interrupt ID port is called “MExternalInterruptID”.

The “deferint” port is an active-high artifact input that, when written to 1, prevents any pending-and-enabled interrupt being taken (normally, such an interrupt would be taken on the next instruction after it becomes pending-and-enabled). The purpose of this signal is to enable alignment with hardware models in step-and-compare usage.
1.10 Debug Mode

The model can be configured to implement Debug mode using parameter “debug_mode”. This implements features described in Chapter 4 of the RISC-V External Debug Support specification with version specified by parameter “debug_version” (see References). Some aspects of this mode are not defined in the specification because they are implementation-specific; the model provides infrastructure to allow implementation of a Debug Module using a custom harness. Features added are described below.

Parameter “debug_mode” can be used to specify three different behaviors, as follows:

1. If set to value “vector”, then operations that would cause entry to Debug mode result in the processor jumping to the address specified by the “debug_address” parameter. It will execute at this address, in Debug mode, until a “dret” instruction causes return to non-Debug mode. Any exception generated during this execution will cause a jump to the address specified by the “dexc_address” parameter.

2. If set to value “interrupt”, then operations that would cause entry to Debug mode result in the processor simulation call (e.g. opProcessorSimulate) returning, with a stop reason of OP_SR_INTERRUPT. In this usage scenario, the Debug Module is implemented in the simulation harness.

3. If set to value “halt”, then operations that would cause entry to Debug mode result in the processor halting. Depending on the simulation environment, this might cause a return from the simulation call with a stop reason of OP_SR_HALT, or debug mode might be implemented by another platform component which then restarts the debugged processor again.

1.10.1 Debug State Entry

The specification does not define how Debug mode is implemented. In this model, Debug mode is enabled by a Boolean pseudo-register, “DM”. When “DM” is True, the processor is in Debug mode. When “DM” is False, mode is defined by “mstatus” in the usual way.

Entry to Debug mode can be performed in any of these ways:

1. By writing True to register “DM” (e.g. using opProcessorRegWrite) followed by simulation of at least one cycle (e.g. using opProcessorSimulate), dcsr cause will be reported as trigger;

2. By writing a 1 then 0 to net “haltreq” (using opNetWrite) followed by simulation of at least one cycle (e.g. using opProcessorSimulate);

3. By writing a 1 to net “resethaltreq” (using opNetWrite) while the “reset” signal undergoes a negedge transition, followed by simulation of at least one cycle (e.g. using opProcessorSimulate);

4. By executing an “ebreak” instruction when Debug mode entry for the current processor mode is enabled by dcsr.ebreakm, dcsr.ebreaks or dcsr.ebreaku.

In all cases, the processor will save required state in “dpc” and “dcsr” and then perform actions described above, depending in the value of the “debug_mode” parameter.
1.10.2 Debug State Exit

Exit from Debug mode can be performed in any of these ways:

1. By writing False to register “DM” (e.g. using opProcessorRegWrite) followed by simulation of at least one cycle (e.g. using opProcessorSimulate);
2. By executing an “dret” instruction when Debug mode.

In both cases, the processor will perform the steps described in section 4.6 (Resume) of the Debug specification.

1.10.3 Debug Registers

When Debug mode is enabled, registers “dcsr”, “dpc”, “dscratch0” and “dscratch1” are implemented as described in the specification. These may be manipulated externally by a Debug Module using opProcessorRegRead or opProcessorRegWrite; for example, the Debug Module could write “dcsr” to enable “ebreak” instruction behavior as described above, or read and write “dpc” to emulate stepping over an “ebreak” instruction prior to resumption from Debug mode.

1.10.4 Debug Mode Execution

The specification allows execution of code fragments in Debug mode. A Debug Module implementation can cause execution in Debug mode by the following steps:

1. Write the address of a Program Buffer to the program counter using opProcessorPCSet;
2. If “debug_mode” is set to “halt”, write 0 to pseudo-register “DMStall” (to leave halted state);
3. If entry to Debug mode was handled by exiting the simulation callback, call opProcessorSimulate or opRootModuleSimulate to resume simulation.

Debug mode will be re-entered in these cases:

1. By execution of an “ebreak” instruction; or:
2. By execution of an instruction that causes an exception.

In both cases, the processor will either jump to the debug exception address, or return control immediately to the harness, with stopReason of OP_SR_INTERRUPT, or perform a halt, depending on the value of the “debug_mode” parameter.

1.10.5 Debug Single Step

When in Debug mode, the processor or harness can cause a single instruction to be executed on return from that mode by setting dcsr.step. After one non-Debug-mode instruction has been executed, control will be returned to the harness. The processor will remain in single-step mode until dcsr.step is cleared.
1.10.6 Debug Event Priorities

The model supports two different models for determining which debug exception occurs when multiple debug events are pending:

1: original mode (when parameter “debug_priority” = “original”);

2: modified mode, as described in Debug Specification pull request 693 (when parameter “debug_priority” = “PR693”). This mode resolves some anomalous behavior of the original specification.

1.10.7 Debug Ports

Port “DM” is an output signal that indicates whether the processor is in Debug mode

Port “haltreq” is a rising-edge-triggered signal that triggers entry to Debug mode (see above).

Port “resethaltreq” is a level-sensitive signal that triggers entry to Debug mode after reset (see above).

1.11 Trigger Module

This model is configured with a trigger module, implementing a subset of the behavior described in Chapter 5 of the RISC-V External Debug Support specification with version specified by parameter “debug_version” (see References).

1.11.1 Trigger Module Restrictions

The model currently supports tdata1 of type 0, type 2 (mcontrol), type 3 (icount), type 4 (ittrigger), type 5 (etrigger) and type 6 (mcontrol6). icount triggers are implemented for a single instruction only, with count hard-wired to 1 and automatic zeroing of mode bits when the trigger fires.

1.11.2 Trigger Module Parameters

Parameter “trigger_num” is used to specify the number of implemented triggers. In this variant, “trigger_num” is 4.

Parameter “tinfo” is used to specify the value of the read-only “tinfo” register, which indicates the trigger types supported. In this variant, “tinfo” is 0x7d.

Parameter “tinfo_undefined” is used to specify whether the “tinfo” register is undefined, in which case reads of it trap to Machine mode. In this variant, “tinfo_undefined” is 0.

Parameter “tcontrol_undefined” is used to specify whether the “tcontrol” register is undefined, in which case accesses to it trap to Machine mode. In this variant, “tcontrol_undefined” is 0.

Parameter “mcontext_undefined” is used to specify whether the “mcontext” register is undefined, in which case accesses to it trap to Machine mode. In this variant, “mcontext_undefined” is 0.
Parameter “scontext_undefined” is used to specify whether the “scontext” register is undefined, in which case accesses to it trap to Machine mode. In this variant, “scontext_undefined” is 0.

Parameter “mscontext_undefined” is used to specify whether the “mscontext” register is undefined, in which case accesses to it trap to Machine mode. In this variant, “mscontext_undefined” is 0.

Parameter “amo_trigger” is used to specify whether load/store triggers are activated for AMO instructions. In this variant, “amo_trigger” is 0.

Parameter “no_hit” is used to specify whether the “hit” bit in tdata1 is unimplemented. In this variant, “no_hit” is 0.

Parameter “no_sselect_2” is used to specify whether the “sselect” field in “textra32”/“textra64” registers is unable to hold value 2 (indicating match by ASID is not allowed). In this variant, “no_sselect_2” is 0.

Parameter “mcontext_bits” is used to specify the number of writable bits in the “mcontext” register. In this variant, “mcontext_bits” is 6.

Parameter “scontext_bits” is used to specify the number of writable bits in the “scontext” register. In this variant, “scontext_bits” is 16.

Parameter “mvalue_bits” is used to specify the number of writable bits in the “mvalue” field in “textra32”/“textra64” registers; if zero, the “mselect” field is tied to zero. In this variant, “mvalue_bits” is 6.

Parameter “svalue_bits” is used to specify the number of writable bits in the “svalue” field in “textra32”/“textra64” registers; if zero, the “sselect” is tied to zero. In this variant, “svalue_bits” is 16.

Parameter “mcontrol_maskmax” is used to specify the value of field “maskmax” in the “mcontrol” register. In this variant, “mcontrol_maskmax” is 63.

### 1.12 Debug Mask

It is possible to enable model debug messages in various categories. This can be done statically using the “debugflags” parameter, or dynamically using the “debugflags” command. Enabled messages are specified using a bitmask value, as follows:

- Value 0x002: enable debugging of PMP and virtual memory state;
- Value 0x004: enable debugging of interrupt state.

All other bits in the debug bitmask are reserved and must not be set to non-zero values.

### 1.13 Integration Support

This model implements a number of non-architectural pseudo-registers and other features to facilitate integration.
1.13.1 CSR Register External Implementation

If parameter “enable_CSR_bus” is True, an artifact 16-bit bus “CSR” is enabled. Slave callbacks installed on this bus can be used to implement modified CSR behavior (use opBusSlaveNew or icmMapExternalMemory, depending on the client API). A CSR with index 0xABC is mapped on the bus at address 0xABC0; as a concrete example, implementing CSR “time” (number 0xC01) externally requires installation of callbacks at address 0xC010 on the CSR bus.

1.13.2 Page Table Walk Introspection

Artifact register “PTWStage” shows the active page table translation stage (0 if no stage active, 1 if HS-stage active, 2 if VS-stage active and 3 if G-stage active). This register is visibly non-zero only in a memory access callback triggered by a page table walk event.

Artifact register “PTWInputAddr” shows the input address of active page table translation. This register is visibly non-zero only in a memory access callback triggered by a page table walk event.

Artifact register “PTWLevel” shows the active level of page table translation (corresponding to index variable “i” in the algorithm described by Virtual Address Translation Process in the RISC-V Privileged Architecture specification). This register is visibly non-zero only in a memory access callback triggered by a page table walk event.

1.14 Limitations

Instruction pipelines are not modeled in any way. All instructions are assumed to complete immediately. This means that instruction barrier instructions (e.g. fence.i) are treated as NOPs, with the exception of any Illegal Instruction behavior, which is modeled.

Caches and write buffers are not modeled in any way. All loads, fetches and stores complete immediately and in order, and are fully synchronous. Data barrier instructions (e.g. fence) are treated as NOPs, with the exception of any Illegal Instruction behavior, which is modeled.

Real-world timing effects are not modeled: all instructions are assumed to complete in a single cycle.

Hardware Performance Monitor registers are not implemented and hardwired to zero.

The TLB is architecturally-accurate but not device accurate. This means that all TLB maintenance and address translation operations are fully implemented but the cache is larger than in the real device.

1.15 Verification

All instructions have been extensively tested by Imperas, using tests generated specifically for this model and also reference tests from https://github.com/riscv/riscv-tests.

Also reference tests have been used from various sources including:
https://github.com/riscv/riscv-tests
https://github.com/ucb-bar/riscv-torture

The Imperas OVPsim RISC-V models are used in the RISC-V Foundation Compliance Framework as a functional Golden Reference:
https://github.com/riscv/riscv-compliance

where the simulated model is used to provide the reference signatures for compliance testing. The Imperas OVPsim RISC-V models are used as reference in both open source and commercial instruction stream test generators for hardware design verification, for example:
http://valtrix.in/sting from Valtrix
https://github.com/google/riscv-dv from Google

The Imperas OVPsim RISC-V models are also used by commercial and open source RISC-V Core RTL developers as a reference to ensure correct functionality of their IP.

1.16 References

The Model details are based upon the following specifications:
RISC-V Instruction Set Manual, Volume II: Privileged Architecture (Privileged Architecture Version 1.12, equivalent to 20211203)
RISC-V External Debug Support (RISC-V External Debug Support Version 1.0.0-STABLE)
Chapter 2

Configuration

2.1 Location

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

2.2 GDB Path

The default GDB for this model is: $IMPERAS_HOME/lib/$IMPERAS_ARCH/gdb/riscv-none-embed-gdb.

2.3 Semi-Host Library

The default semi-host library file is riscv.ovpworld.org/semihosting/pk/1.0

2.4 Processor Endian-ness

This is a LITTLE 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: 0xf3.
Chapter 3

All Variants in this model

This model has these variants

<table>
<thead>
<tr>
<th>Variant</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RV32I</td>
<td></td>
</tr>
<tr>
<td>RV32IM</td>
<td>(described in this document)</td>
</tr>
<tr>
<td>RV32IMC</td>
<td></td>
</tr>
<tr>
<td>RV32IMCZce</td>
<td></td>
</tr>
<tr>
<td>RV32IMAC</td>
<td></td>
</tr>
<tr>
<td>RV32G</td>
<td></td>
</tr>
<tr>
<td>RV32GC</td>
<td></td>
</tr>
<tr>
<td>RV32GCZfinx</td>
<td></td>
</tr>
<tr>
<td>RV32GCB</td>
<td></td>
</tr>
<tr>
<td>RV32GCH</td>
<td></td>
</tr>
<tr>
<td>RV32GCK</td>
<td></td>
</tr>
<tr>
<td>RV32GCN</td>
<td></td>
</tr>
<tr>
<td>RV32GCP</td>
<td></td>
</tr>
<tr>
<td>RV32GCV</td>
<td></td>
</tr>
<tr>
<td>RV32E</td>
<td></td>
</tr>
<tr>
<td>RV32EC</td>
<td></td>
</tr>
<tr>
<td>RV32EM</td>
<td></td>
</tr>
<tr>
<td>RV64I</td>
<td></td>
</tr>
<tr>
<td>RV64IM</td>
<td></td>
</tr>
<tr>
<td>RV64IMC</td>
<td></td>
</tr>
<tr>
<td>RV64IMCZce</td>
<td></td>
</tr>
<tr>
<td>RV64IMAC</td>
<td></td>
</tr>
<tr>
<td>RV64G</td>
<td></td>
</tr>
<tr>
<td>RV64GC</td>
<td></td>
</tr>
<tr>
<td>RV64GCZfinx</td>
<td></td>
</tr>
<tr>
<td>RV64GCB</td>
<td></td>
</tr>
<tr>
<td>RV64GCH</td>
<td></td>
</tr>
<tr>
<td>RV64GCK</td>
<td></td>
</tr>
<tr>
<td>RV64GCN</td>
<td></td>
</tr>
<tr>
<td>RV64GCP</td>
<td></td>
</tr>
<tr>
<td>RV64GCV</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>-------</td>
<td>-------</td>
</tr>
<tr>
<td>RVB32I</td>
<td></td>
</tr>
<tr>
<td>RVB32E</td>
<td></td>
</tr>
<tr>
<td>RVB64I</td>
<td></td>
</tr>
</tbody>
</table>

Table 3.1: All Variants in this model
Chapter 4

Bus Master Ports

This model has these bus master ports.

<table>
<thead>
<tr>
<th>Name</th>
<th>min</th>
<th>max</th>
<th>Connect?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>INSTRUCTION</td>
<td>32</td>
<td>34</td>
<td>mandatory</td>
<td>Instruction bus</td>
</tr>
<tr>
<td>DATA</td>
<td>32</td>
<td>34</td>
<td>optional</td>
<td>Data bus</td>
</tr>
</tbody>
</table>

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.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Connect?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>reset</td>
<td>input</td>
<td>optional</td>
<td>Reset</td>
</tr>
<tr>
<td>reset_addr</td>
<td>input</td>
<td>optional</td>
<td>externally-applied reset address</td>
</tr>
<tr>
<td>nmi</td>
<td>input</td>
<td>optional</td>
<td>NMI</td>
</tr>
<tr>
<td>nmi_cause</td>
<td>input</td>
<td>optional</td>
<td>externally-applied NMI cause</td>
</tr>
<tr>
<td>nmi_addr</td>
<td>input</td>
<td>optional</td>
<td>externally-applied NMI address</td>
</tr>
<tr>
<td>SSWInterrupt</td>
<td>input</td>
<td>optional</td>
<td>Supervisor software interrupt</td>
</tr>
<tr>
<td>MSWInterrupt</td>
<td>input</td>
<td>optional</td>
<td>Machine software interrupt</td>
</tr>
<tr>
<td>STimerInterrupt</td>
<td>input</td>
<td>optional</td>
<td>Supervisor timer interrupt</td>
</tr>
<tr>
<td>MTimerInterrupt</td>
<td>input</td>
<td>optional</td>
<td>Machine timer interrupt</td>
</tr>
<tr>
<td>SExternalInterrupt</td>
<td>input</td>
<td>optional</td>
<td>Supervisor external interrupt</td>
</tr>
<tr>
<td>MExternalInterrupt</td>
<td>input</td>
<td>optional</td>
<td>Machine external interrupt</td>
</tr>
<tr>
<td>irq_ack_o</td>
<td>output</td>
<td>optional</td>
<td>interrupt acknowledgment (pulse)</td>
</tr>
<tr>
<td>irq_id_o</td>
<td>output</td>
<td>optional</td>
<td>acknowledged interrupt id (valid during irq_ack_o pulse)</td>
</tr>
<tr>
<td>sec_lvl_o</td>
<td>output</td>
<td>optional</td>
<td>current privilege level</td>
</tr>
<tr>
<td>deferint</td>
<td>input</td>
<td>optional</td>
<td>Artifact signal causing interrupts to be held off when high</td>
</tr>
</tbody>
</table>

Table 6.1: Net Ports

21
Chapter 7

FIFO Ports

This model has no FIFO ports.
Chapter 8

Formal Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Fundamental</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>variant</td>
<td>Enumeration</td>
<td>Selects variant (either a generic UISA or a specific model)</td>
</tr>
<tr>
<td>user_version</td>
<td>Enumeration</td>
<td>Specify required User Architecture version (2.2, 2.3, 20190305 or 20191213)</td>
</tr>
<tr>
<td>priv_version</td>
<td>Enumeration</td>
<td>Specify required Privileged Architecture version (1.10, 1.11, 20190405, 20190608, 20211203, 1.12 or master)</td>
</tr>
<tr>
<td>Smepmp_version</td>
<td>Enumeration</td>
<td>Specify required Smepmp Architecture version (none, 0.9.5 or 1.0)</td>
</tr>
<tr>
<td>numHarts</td>
<td>Uns32</td>
<td>Specify the number of hart contexts in a multiprocessor</td>
</tr>
<tr>
<td>endian</td>
<td>Endian</td>
<td>Model endian</td>
</tr>
<tr>
<td>enable_expanded</td>
<td>Boolean</td>
<td>Specify that 48-bit and 64-bit expanded instructions are supported</td>
</tr>
<tr>
<td>endianFixed</td>
<td>Boolean</td>
<td>Specify that data endianness is fixed (mstatus.{MBE,SBE,UBE} fields are read-only)</td>
</tr>
<tr>
<td>misa_MXL</td>
<td>Uns32</td>
<td>Override default value of misa.MXL</td>
</tr>
<tr>
<td>misa_Extensions</td>
<td>Uns32</td>
<td>Override default value of misa.Extensions</td>
</tr>
<tr>
<td>add_Extensions</td>
<td>String</td>
<td>Add extensions specified by letters to misa.Extensions (for example, specify “VD” to add V and D features)</td>
</tr>
<tr>
<td>sub_Extensions</td>
<td>String</td>
<td>Remove extensions specified by letters from misa.Extensions (for example, specify “VD” to remove V and D features)</td>
</tr>
<tr>
<td>misa_Extensions_mask</td>
<td>Uns32</td>
<td>Override mask of writable bits in misa.Extensions</td>
</tr>
<tr>
<td>add_Extensions_mask</td>
<td>String</td>
<td>Add extensions specified by letters to mask of writable bits in misa.Extensions (for example, specify “VD” to add V and D features)</td>
</tr>
<tr>
<td>sub_Extensions_mask</td>
<td>String</td>
<td>Remove extensions specified by letters from mask of writable bits in misa.Extensions (for example, specify “VD” to remove V and D features)</td>
</tr>
<tr>
<td>add_implicit_Extensions</td>
<td>String</td>
<td>Add extensions specified by letters to implicitly-present extensions not visible in misa.Extensions</td>
</tr>
<tr>
<td>sub_implicit_Extensions</td>
<td>String</td>
<td>Remove extensions specified by letters from implicitly-present extensions not visible in misa.Extensions</td>
</tr>
<tr>
<td><strong>Compressed Extension</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>compress_version</td>
<td>Enumeration</td>
<td>Specify required Compressed Architecture version (legacy, 0.70.1 or 0.70.5)</td>
</tr>
<tr>
<td><strong>Debug Extension</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>debug_version</td>
<td>Enumeration</td>
<td>Specify required Debug Architecture version (0.13.2-DRAFT, 0.14.0-DRAFT or 1.0.0-STABLE)</td>
</tr>
<tr>
<td>debug_mode</td>
<td>Enumeration</td>
<td>Specify how Debug mode is implemented (none, vector, interrupt or halt)</td>
</tr>
<tr>
<td><strong>Interrupts_Exceptions</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>rnmi_version</td>
<td>Enumeration</td>
<td>Specify required RNMI Architecture version (none or 0.2.1)</td>
</tr>
<tr>
<td>mtvec_is_ro</td>
<td>Boolean</td>
<td>Specify whether mtvec CSR is read-only</td>
</tr>
<tr>
<td>tvec_align</td>
<td>Uns32</td>
<td>Specify hardware-enforced alignment of mtvec/stvec/utvec when Vectored interrupt mode enabled</td>
</tr>
<tr>
<td>ecode_mask</td>
<td>Uns64</td>
<td>Specify hardware-enforced mask of writable bits in xcause.ExceptionCode</td>
</tr>
<tr>
<td><strong>e_code_nmi</strong></td>
<td><strong>Uns64</strong></td>
<td>Specify xcause.ExceptionCode for NMI</td>
</tr>
<tr>
<td>---------------</td>
<td>-----------</td>
<td>-------------------------------------</td>
</tr>
<tr>
<td><strong>nmi_is_latched</strong></td>
<td><strong>Boolean</strong></td>
<td>Specify whether NMI input is latched on rising edge (if False, it is level-sensitive)</td>
</tr>
<tr>
<td><strong>tval_zero</strong></td>
<td><strong>Boolean</strong></td>
<td>Specify whether mtval/stval/utval are hard wired to zero</td>
</tr>
<tr>
<td><strong>tval_zero_esebreak</strong></td>
<td><strong>Boolean</strong></td>
<td>Specify whether mtval/stval/utval are set to zero by an ebreak</td>
</tr>
<tr>
<td><strong>tval_ui_code</strong></td>
<td><strong>Boolean</strong></td>
<td>Specify whether mtval/stval contain faulting instruction bits on illegal instruction exception</td>
</tr>
<tr>
<td><strong>reset_address</strong></td>
<td><strong>Uns64</strong></td>
<td>Override reset vector address</td>
</tr>
<tr>
<td><strong>nmi_address</strong></td>
<td><strong>Uns64</strong></td>
<td>Override NMI vector address</td>
</tr>
<tr>
<td><strong>CLINT_address</strong></td>
<td><strong>Uns64</strong></td>
<td>Specify base address of internal CLINT model (or 0 for no CLINT)</td>
</tr>
<tr>
<td><strong>local_int_num</strong></td>
<td><strong>Uns32</strong></td>
<td>Specify number of supplemental local interrupts</td>
</tr>
<tr>
<td><strong>unimp_int_mask</strong></td>
<td><strong>Uns64</strong></td>
<td>Specify mask of unimplemented interrupts (e.g. 1&lt;&lt;9 indicates Supervisor external interrupt unimplemented)</td>
</tr>
<tr>
<td><strong>force_mideleg</strong></td>
<td><strong>Uns64</strong></td>
<td>Specify mask of interrupts always delegated to lower-priority execution level from Machine execution level</td>
</tr>
<tr>
<td><strong>force_sideleg</strong></td>
<td><strong>Uns64</strong></td>
<td>Specify mask of interrupts always delegated to User execution level from Supervisor execution level</td>
</tr>
<tr>
<td><strong>no_ideleg</strong></td>
<td><strong>Uns64</strong></td>
<td>Specify mask of interrupts that cannot be delegated to lower-priority execution levels</td>
</tr>
<tr>
<td><strong>no_edeleg</strong></td>
<td><strong>Uns64</strong></td>
<td>Specify mask of exceptions that cannot be delegated to lower-priority execution levels</td>
</tr>
<tr>
<td><strong>external_int_id</strong></td>
<td><strong>Boolean</strong></td>
<td>Whether to add nets allowing External Interrupt ID codes to be forced</td>
</tr>
</tbody>
</table>

**SimulationArtifact**

| **use_hw_reg_names** | **Boolean** | Specify whether to use hardware register names x0-x31 and f0-f31 instead of ABI register names |
| **no_pseudo_inst** | **Boolean** | Specify whether pseudo-instructions should not be reported in trace and disassembly |
| **verbose** | **Boolean** | Specify verbose output messages |
| **traceVolatile** | **Boolean** | Specify whether volatile registers (e.g. minstret) should be shown in change trace |
| **enable_CSR_bus** | **Boolean** | Add artifact CSR bus port, allowing CSR registers to be externally implemented |
| **CSR_remap** | **String** | Comma-separated list of CSR number mappings, each of the form <csr-Name>=<number> |
| **ASID_cache_size** | **Uns32** | Specifies the number of different ASIDs for which TLB entries are cached; a value of 0 implies no limit |

**Memory**

| **updatePTEA** | **Boolean** | Specify whether hardware update of PTE A bit is supported |
| **updatePTED** | **Boolean** | Specify whether hardware update of PTE D bit is supported |
| **unaligned_low_pri** | **Boolean** | Specify whether address misaligned exceptions are lower priority than page or access fault exceptions |
| **unaligned** | **Boolean** | Specify whether the processor supports unaligned memory accesses |
| **ASID_bits** | **Uns32** | Specify the number of implemented ASID bits |
| **Sv_modes** | **Uns32** | Specify bit mask of implemented address translation modes (e.g. (1<<0)+(1<<8) indicates “bare” and “Sv39” modes may be selected in satp.MODE) |

**Instruction_CSR_Behavior**

<p>| <strong>wh_is_nop</strong> | <strong>Boolean</strong> | Specify whether WFI should be treated as a NOP (if not, halt while waiting for interrupts) |
| <strong>counteren_mask</strong> | <strong>Uns32</strong> | Specify hardware-enforced mask of writable bits in mcounteren/scounteren registers |
| <strong>noinhibit_mask</strong> | <strong>Uns32</strong> | Specify hardware-enforced mask of always-zero bits in mcounteren/scounteren registers |
| <strong>cycle_undefined</strong> | <strong>Boolean</strong> | Specify that the cycle CSR is undefined |
| <strong>mcycle_undefined</strong> | <strong>Boolean</strong> | Specify that the mcycle CSR is undefined |
| <strong>time_undefined</strong> | <strong>Boolean</strong> | Specify that the time CSR is undefined |</p>
<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>instret_undefined</td>
<td>Boolean</td>
<td>Specify that the instret CSR is undefined</td>
</tr>
<tr>
<td>minstret_undefined</td>
<td>Boolean</td>
<td>Specify that the minstret CSR is undefined</td>
</tr>
<tr>
<td>hpmcounter_undefined</td>
<td>Boolean</td>
<td>Specify that the hpmcounter CSRs are undefined</td>
</tr>
<tr>
<td>mhpmcounter_undefined</td>
<td>Boolean</td>
<td>Specify that the mhpmcounter CSRs are undefined</td>
</tr>
<tr>
<td>CSR_Masks</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mtvec_mask</td>
<td>Uns64</td>
<td>Specify hardware-enforced mask of writable bits in mtvec register</td>
</tr>
<tr>
<td>stvec_mask</td>
<td>Uns64</td>
<td>Specify hardware-enforced mask of writable bits in stvec register</td>
</tr>
<tr>
<td>tdata1_mask</td>
<td>Uns64</td>
<td>Specify hardware-enforced mask of writable bits in Trigger Module tdata1</td>
</tr>
<tr>
<td>mip_mask</td>
<td>Uns64</td>
<td>Specify hardware-enforced mask of writable bits in mip register</td>
</tr>
<tr>
<td>sip_mask</td>
<td>Uns64</td>
<td>Specify hardware-enforced mask of writable bits in sip register</td>
</tr>
<tr>
<td>envcfg_mask</td>
<td>Uns64</td>
<td>Specify hardware-enforced mask of writable bits in envcfg registers</td>
</tr>
<tr>
<td>mtvec_sext</td>
<td>Boolean</td>
<td>Specify whether mtvec is sign-extended from most-significant bit</td>
</tr>
<tr>
<td>stvec_sext</td>
<td>Boolean</td>
<td>Specify whether stvec is sign-extended from most-significant bit</td>
</tr>
<tr>
<td>Trigger</td>
<td></td>
<td></td>
</tr>
<tr>
<td>tinfo_undefined</td>
<td>Boolean</td>
<td>Specify that the tinfo CSR is undefined</td>
</tr>
<tr>
<td>tcontrol_undefined</td>
<td>Boolean</td>
<td>Specify that the tcontrol CSR is undefined</td>
</tr>
<tr>
<td>mcontext_undefined</td>
<td>Boolean</td>
<td>Specify that the mcontext CSR is undefined</td>
</tr>
<tr>
<td>scontext_undefined</td>
<td>Boolean</td>
<td>Specify that the scontext CSR is undefined</td>
</tr>
<tr>
<td>mscontext_undefined</td>
<td>Boolean</td>
<td>Specify that the mscontext CSR is undefined</td>
</tr>
<tr>
<td>amo_trigger</td>
<td>Boolean</td>
<td>Specify whether AMO load/store operations activate triggers</td>
</tr>
<tr>
<td>no_hit</td>
<td>Boolean</td>
<td>Specify that tdata1.hit is unimplemented</td>
</tr>
<tr>
<td>no_sselect_2</td>
<td>Boolean</td>
<td>Specify that textra.sselect=2 is not supported (no trigger match by ASID)</td>
</tr>
<tr>
<td>trigger_num</td>
<td>Uns32</td>
<td>Specify the number of implemented hardware triggers</td>
</tr>
<tr>
<td>tinfo</td>
<td>Uns32</td>
<td>Override tinfo register (for all triggers)</td>
</tr>
<tr>
<td>mcontext_bits</td>
<td>Uns32</td>
<td>Specify the number of implemented bits in mcontext</td>
</tr>
<tr>
<td>scontext_bits</td>
<td>Uns32</td>
<td>Specify the number of implemented bits in scontext</td>
</tr>
<tr>
<td>mvalue_bits</td>
<td>Uns32</td>
<td>Specify the number of implemented bits in textra.mvalue (if zero, tex-</td>
</tr>
<tr>
<td></td>
<td></td>
<td>tra.mselect is tied to zero)</td>
</tr>
<tr>
<td>svalue_bits</td>
<td>Uns32</td>
<td>Specify the number of implemented bits in textra.svalue (if zero, tex-</td>
</tr>
<tr>
<td></td>
<td></td>
<td>tra.sselect is tied to zero)</td>
</tr>
<tr>
<td>mcontrol_maskmax</td>
<td>Uns32</td>
<td>Specify mcontrol.maskmax value</td>
</tr>
<tr>
<td>PMP_Configuration</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PMP_grain</td>
<td>Uns32</td>
<td>Specify PMP region granularity, G (0 = &gt;4 bytes, 1 = &gt;8 bytes, etc)</td>
</tr>
<tr>
<td>PMP_registers</td>
<td>Uns32</td>
<td>Specify the number of implemented PMP address registers</td>
</tr>
<tr>
<td>PMP_max_page</td>
<td>Uns32</td>
<td>Specify the maximum size of PMP region to map if non-zero (may improve</td>
</tr>
<tr>
<td></td>
<td></td>
<td>performance; constrained to a power of two)</td>
</tr>
<tr>
<td>PMP_decompose</td>
<td>Boolean</td>
<td>Whether unaligned PMP accesses are decomposed into separate aligned</td>
</tr>
<tr>
<td></td>
<td></td>
<td>accesses</td>
</tr>
<tr>
<td>PMP_undefined</td>
<td>Boolean</td>
<td>Whether accesses to unimplemented PMP registers are undefined (if True)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>or write ignored and zero (if False)</td>
</tr>
<tr>
<td>PMP_maskparams</td>
<td>Boolean</td>
<td>Enable parameters to change the read-only masks for PMP CSRs</td>
</tr>
<tr>
<td>PMP_initialparams</td>
<td>Boolean</td>
<td>Enable parameters to change the reset values for PMP CSRs</td>
</tr>
<tr>
<td>Other_Extensions</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Svnapot_page_mask</td>
<td>Uns64</td>
<td>Specify mask of implemented Svnapot intermediate page sizes (e.g. 1&lt;&lt;16</td>
</tr>
<tr>
<td></td>
<td></td>
<td>means 64KiB contiguous regions are supported)</td>
</tr>
<tr>
<td>Smastateen</td>
<td>Boolean</td>
<td>Specify that Smastateen is implemented</td>
</tr>
<tr>
<td>Svpbmt</td>
<td>Boolean</td>
<td>Specify that Svpbmt is implemented</td>
</tr>
<tr>
<td>Svinval</td>
<td>Boolean</td>
<td>Specify that Svinval is implemented</td>
</tr>
<tr>
<td>Zicsr</td>
<td>Boolean</td>
<td>Specify that Zicsr is implemented</td>
</tr>
<tr>
<td>Zifenceci</td>
<td>Boolean</td>
<td>Specify that Zifenceci is implemented</td>
</tr>
<tr>
<td>Ziboom</td>
<td>Boolean</td>
<td>Specify that Ziboom is implemented</td>
</tr>
<tr>
<td>Zibop</td>
<td>Boolean</td>
<td>Specify that Zibop is implemented</td>
</tr>
<tr>
<td>Zicboz</td>
<td>Boolean</td>
<td>Specify that Zicboz is implemented</td>
</tr>
<tr>
<td>Zmmul</td>
<td>Boolean</td>
<td>Specify that Zmmul is implemented</td>
</tr>
</tbody>
</table>

Copyright (c) 2022 Imperas Software Limited
OVP License. Release 20220722.0
<table>
<thead>
<tr>
<th>CSR_defaults</th>
<th>mvendorid</th>
<th>Uns64</th>
<th>Override mvendorid register</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>marchid</td>
<td>Uns64</td>
<td>Override marchid register</td>
</tr>
<tr>
<td></td>
<td>mimpid</td>
<td>Uns64</td>
<td>Override mimpid register</td>
</tr>
<tr>
<td></td>
<td>mhartid</td>
<td>Uns64</td>
<td>Override mhartid register  (or first mhartid of an incrementing sequence if this is an SMP variant)</td>
</tr>
<tr>
<td></td>
<td>mconfigptr</td>
<td>Uns64</td>
<td>Override mconfigptr register</td>
</tr>
<tr>
<td></td>
<td>mtvec</td>
<td>Uns64</td>
<td>Override mtvec register</td>
</tr>
<tr>
<td></td>
<td>mseccfg</td>
<td>Uns64</td>
<td>Override mseccfg register</td>
</tr>
</tbody>
</table>

| Floating_Point | mstatus_FS_zero | Boolean | Specify that mstatus.FS is hard-wired to zero |
|                |                |         |                                              |
|                | mtvec           |         | Specify number of interrupt levels implemented by CLIC, or 0 if CLIC absent |

| Table 8.1: Parameters that can be set in: Hart |

8.1 Parameters with enumerated types

8.1.1 Parameter user_version

<table>
<thead>
<tr>
<th>Set to this value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.2</td>
<td>User Architecture Version 2.2</td>
</tr>
<tr>
<td>2.3</td>
<td>Deprecated and equivalent to 20191213</td>
</tr>
<tr>
<td>20190305</td>
<td>Deprecated and equivalent to 20191213</td>
</tr>
<tr>
<td>20191213</td>
<td>User Architecture Version 20191213</td>
</tr>
</tbody>
</table>

| Table 8.2: Values for Parameter user_version |

8.1.2 Parameter priv_version

<table>
<thead>
<tr>
<th>Set to this value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.10</td>
<td>Privileged Architecture Version 1.10</td>
</tr>
<tr>
<td>1.11</td>
<td>Privileged Architecture Version 1.11, equivalent to 20190608</td>
</tr>
<tr>
<td>20190405</td>
<td>Deprecated and equivalent to 20190608</td>
</tr>
<tr>
<td>20190608</td>
<td>Privileged Architecture Version Ratified-IMFDQC-and-Priv-v1.11</td>
</tr>
<tr>
<td>20211203</td>
<td>Privileged Architecture Version 20211203</td>
</tr>
<tr>
<td>1.12</td>
<td>Privileged Architecture Version 1.12, equivalent to 20211203</td>
</tr>
<tr>
<td>master</td>
<td>Privileged Architecture Master Branch as of commit 6bdeb58 (this is subject to change)</td>
</tr>
</tbody>
</table>

| Table 8.3: Values for Parameter priv_version |

8.1.3 Parameter compress_version

<table>
<thead>
<tr>
<th>Set to this value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>legacy</td>
<td>Compressed Architecture absent or legacy version</td>
</tr>
<tr>
<td>0.70.1</td>
<td>Compressed Architecture Version 0.70.1</td>
</tr>
<tr>
<td>0.70.5</td>
<td>Compressed Architecture Version 0.70.5</td>
</tr>
</tbody>
</table>

| Table 8.4: Values for Parameter compress_version |

8.1.4 Parameter debug_version
Table 8.5: Values for Parameter debug_version

<table>
<thead>
<tr>
<th>Set to this value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.13.2-DRAFT</td>
<td>RISC-V External Debug Support Version 0.13.2-DRAFT</td>
</tr>
<tr>
<td>0.14.0-DRAFT</td>
<td>RISC-V External Debug Support Version 0.14.0-DRAFT</td>
</tr>
<tr>
<td>1.0.0-STABLE</td>
<td>RISC-V External Debug Support Version 1.0.0-STABLE</td>
</tr>
</tbody>
</table>

8.1.5 Parameter rnmi_version

<table>
<thead>
<tr>
<th>Set to this value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td>RNMI not implemented</td>
</tr>
<tr>
<td>0.2.1</td>
<td>RNMI version 0.2.1</td>
</tr>
</tbody>
</table>

Table 8.6: Values for Parameter rnmi_version

8.1.6 Parameter Smepmp_version

<table>
<thead>
<tr>
<th>Set to this value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td>Smepmp not implemented</td>
</tr>
<tr>
<td>0.9.5</td>
<td>Smepmp version 0.9.5 (deprecated and identical to 1.0)</td>
</tr>
<tr>
<td>1.0</td>
<td>Smepmp version 1.0</td>
</tr>
</tbody>
</table>

Table 8.7: Values for Parameter Smepmp_version

8.1.7 Parameter debug_mode

<table>
<thead>
<tr>
<th>Set to this value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td>Debug mode not implemented</td>
</tr>
<tr>
<td>vector</td>
<td>Debug mode implemented by execution at vector</td>
</tr>
<tr>
<td>interrupt</td>
<td>Debug mode implemented by interrupt</td>
</tr>
<tr>
<td>halt</td>
<td>Debug mode implemented by halt</td>
</tr>
</tbody>
</table>

Table 8.8: Values for Parameter debug_mode

8.2 Parameter values

These are the current parameter values.

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fundamental</td>
<td></td>
</tr>
<tr>
<td>variant</td>
<td>RV32IM</td>
</tr>
<tr>
<td>user_version</td>
<td>20191213</td>
</tr>
<tr>
<td>priv_version</td>
<td>1.12</td>
</tr>
<tr>
<td>Smepmp_version</td>
<td>none</td>
</tr>
<tr>
<td>numHarts</td>
<td>0</td>
</tr>
<tr>
<td>endian</td>
<td>none</td>
</tr>
<tr>
<td>enable_expanded</td>
<td>F</td>
</tr>
<tr>
<td>endianFixed</td>
<td>F</td>
</tr>
<tr>
<td>misa_MXL</td>
<td>1</td>
</tr>
<tr>
<td>misa_Extensions</td>
<td>0x141100</td>
</tr>
<tr>
<td>Description</td>
<td>Value</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>add Extensions</td>
<td></td>
</tr>
<tr>
<td>sub Extensions</td>
<td></td>
</tr>
<tr>
<td>misa Extensions mask</td>
<td>0x1100</td>
</tr>
<tr>
<td>add Extensions mask</td>
<td></td>
</tr>
<tr>
<td>sub Extensions mask</td>
<td></td>
</tr>
<tr>
<td>add implicit Extensions</td>
<td></td>
</tr>
<tr>
<td>sub implicit Extensions</td>
<td></td>
</tr>
<tr>
<td>Compressed Extension</td>
<td></td>
</tr>
<tr>
<td>compress_version</td>
<td>0.70.5</td>
</tr>
<tr>
<td>Debug Extension</td>
<td></td>
</tr>
<tr>
<td>debug_version</td>
<td>1.0.0-STABLE</td>
</tr>
<tr>
<td>debug_mode</td>
<td>none</td>
</tr>
<tr>
<td>Interrupts Exceptions</td>
<td></td>
</tr>
<tr>
<td>rnmii_version</td>
<td>none</td>
</tr>
<tr>
<td>mtevec_is_ro</td>
<td>F</td>
</tr>
<tr>
<td>tvec_align</td>
<td>0</td>
</tr>
<tr>
<td>ecode_mask</td>
<td>0x7fffffff</td>
</tr>
<tr>
<td>ecode_nmi</td>
<td>0</td>
</tr>
<tr>
<td>nmi_is_latched</td>
<td>F</td>
</tr>
<tr>
<td>tval_zero</td>
<td>F</td>
</tr>
<tr>
<td>tval_zero_ebreak</td>
<td>F</td>
</tr>
<tr>
<td>tval_i_i_code</td>
<td>T</td>
</tr>
<tr>
<td>reset_address</td>
<td>0</td>
</tr>
<tr>
<td>nmi_address</td>
<td>0</td>
</tr>
<tr>
<td>CLINT_address</td>
<td>0</td>
</tr>
<tr>
<td>local_int_num</td>
<td>0</td>
</tr>
<tr>
<td>unimp_int_mask</td>
<td>0</td>
</tr>
<tr>
<td>force_mideleg</td>
<td>0</td>
</tr>
<tr>
<td>force_sideleg</td>
<td>0</td>
</tr>
<tr>
<td>no_idleg</td>
<td>0</td>
</tr>
<tr>
<td>no_eedeg</td>
<td>0</td>
</tr>
<tr>
<td>external_int_id</td>
<td>F</td>
</tr>
<tr>
<td>Simulation Artifact</td>
<td></td>
</tr>
<tr>
<td>use_hw_reg_names</td>
<td>F</td>
</tr>
<tr>
<td>no_pseudo_inst</td>
<td>F</td>
</tr>
<tr>
<td>verbose</td>
<td>F</td>
</tr>
<tr>
<td>traceVolatile</td>
<td>F</td>
</tr>
<tr>
<td>enable_CSR_bus</td>
<td>F</td>
</tr>
<tr>
<td>CSR_remap</td>
<td></td>
</tr>
<tr>
<td>ASID_cache_size</td>
<td>8</td>
</tr>
<tr>
<td>Memory</td>
<td></td>
</tr>
<tr>
<td>updatePTEA</td>
<td>F</td>
</tr>
<tr>
<td>updatePTED</td>
<td>F</td>
</tr>
<tr>
<td>unaligned_low_pri</td>
<td>F</td>
</tr>
<tr>
<td>unaligned</td>
<td>F</td>
</tr>
</tbody>
</table>

Copyright (c) 2022 Imperas Software Limited  www.ovpworld.org
OVP License. Release 20220722.0  Page 28 of 42
| **ASID_bits** | 9 |
| **Sv_nodes** | 3 |

### Instruction_CSR_Behavior

| **wfi_is_nop** | F |
| **counteren_mask** | 0xffffffff |
| **noinhibit_mask** | 0 |
| **cycle_undefined** | F |
| **mcycle_undefined** | F |
| **time_undefined** | F |
| **instret_undefined** | F |
| **minstret_undefined** | F |
| **hpmcounter_undefined** | F |
| **mhpmcounter_undefined** | F |

### CSR_Masks

| **mtvec_mask** | 0 |
| **stvec_mask** | 0 |
| **tdata1_mask** | 0xffffffffffffffff |
| **mip_mask** | 0x337 |
| **sip_mask** | 0x103 |
| **envcfg_mask** | 0 |
| **mtvec_sext** | F |
| **stvec_sext** | F |

### Trigger

| **tinfo_undefined** | F |
| **tcontrol_undefined** | F |
| **mcontext_undefined** | F |
| **scontext_undefined** | F |
| **mscontext_undefined** | F |
| **amo_trigger** | F |
| **no_hit** | F |
| **no_sselect_2** | F |
| **trigger_num** | 4 |
| **tinfo** | 125 |
| **mcontext_bits** | 6 |
| **scontext_bits** | 16 |
| **mvalue_bits** | 6 |
| **svalue_bits** | 16 |
| **mcontrol_maskmax** | 63 |

### PMP Configuration

<p>| <strong>PMP_grain</strong> | 0 |
| <strong>PMP_registers</strong> | 16 |
| <strong>PMP_max_page</strong> | 0 |
| <strong>PMP_decompose</strong> | F |
| <strong>PMP_undefined</strong> | F |
| <strong>PMP_maskparams</strong> | F |</p>
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>PMP_initialparams</td>
<td>F</td>
</tr>
<tr>
<td><strong>Other Extensions</strong></td>
<td></td>
</tr>
<tr>
<td>Svnapot_page_mask</td>
<td>0</td>
</tr>
<tr>
<td>Smstateen</td>
<td>F</td>
</tr>
<tr>
<td>Svpbmt</td>
<td>F</td>
</tr>
<tr>
<td>Svinval</td>
<td>F</td>
</tr>
<tr>
<td>Zicsr</td>
<td>T</td>
</tr>
<tr>
<td>Zifencei</td>
<td>T</td>
</tr>
<tr>
<td>Zicbom</td>
<td>F</td>
</tr>
<tr>
<td>Zicbop</td>
<td>F</td>
</tr>
<tr>
<td>Zicboz</td>
<td>F</td>
</tr>
<tr>
<td>Zmmul</td>
<td>F</td>
</tr>
<tr>
<td><strong>CSR_Defaults</strong></td>
<td></td>
</tr>
<tr>
<td>mvendorid</td>
<td>0</td>
</tr>
<tr>
<td>marchid</td>
<td>0</td>
</tr>
<tr>
<td>mimpid</td>
<td>0</td>
</tr>
<tr>
<td>mhartid</td>
<td>0</td>
</tr>
<tr>
<td>mconfigptr</td>
<td>0</td>
</tr>
<tr>
<td>mtvec</td>
<td>0</td>
</tr>
<tr>
<td>mseccfg</td>
<td>0</td>
</tr>
<tr>
<td><strong>Floating_Point</strong></td>
<td></td>
</tr>
<tr>
<td>mstatus_FS_zero</td>
<td>F</td>
</tr>
<tr>
<td><strong>Fast_Interrupt</strong></td>
<td></td>
</tr>
<tr>
<td>CLICLEVELS</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 8.9: Parameter values
Chapter 9

Execution Modes

<table>
<thead>
<tr>
<th>Mode</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User</td>
<td>0</td>
<td>User mode</td>
</tr>
<tr>
<td>Supervisor</td>
<td>1</td>
<td>Supervisor mode</td>
</tr>
<tr>
<td>Machine</td>
<td>3</td>
<td>Machine mode</td>
</tr>
</tbody>
</table>

Table 9.1: Modes implemented in: Hart
Chapter 10

Exceptions

<table>
<thead>
<tr>
<th>Exception</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>InstructionAddressMisaligned</td>
<td>0</td>
<td>Fetch from unaligned address</td>
</tr>
<tr>
<td>InstructionAccessFault</td>
<td>1</td>
<td>No access permission for fetch</td>
</tr>
<tr>
<td>IllegalInstruction</td>
<td>2</td>
<td>Undecoded, unimplemented or disabled instruction</td>
</tr>
<tr>
<td>Breakpoint</td>
<td>3</td>
<td>EBREAK instruction executed</td>
</tr>
<tr>
<td>LoadAddressMisaligned</td>
<td>4</td>
<td>Load from unaligned address</td>
</tr>
<tr>
<td>LoadAccessFault</td>
<td>5</td>
<td>No access permission for load</td>
</tr>
<tr>
<td>StoreAMOAddressMisaligned</td>
<td>6</td>
<td>Store/atomic memory operation at unaligned address</td>
</tr>
<tr>
<td>StoreAMOAccessFault</td>
<td>7</td>
<td>No access permission for store/atomic memory operation</td>
</tr>
<tr>
<td>EnvironmentCallFromUMode</td>
<td>8</td>
<td>ECALL instruction executed in User mode</td>
</tr>
<tr>
<td>EnvironmentCallFromSMode</td>
<td>9</td>
<td>ECALL instruction executed in Supervisor mode</td>
</tr>
<tr>
<td>EnvironmentCallFromMMode</td>
<td>11</td>
<td>ECALL instruction executed in Machine mode</td>
</tr>
<tr>
<td>InstructionPageFault</td>
<td>12</td>
<td>Page fault at fetch address</td>
</tr>
<tr>
<td>LoadPageFault</td>
<td>13</td>
<td>Page fault at load address</td>
</tr>
<tr>
<td>StoreAMOPageFault</td>
<td>15</td>
<td>Page fault at store/atomic memory operation address</td>
</tr>
<tr>
<td>SSWInterrupt</td>
<td>65</td>
<td>Supervisor software interrupt</td>
</tr>
<tr>
<td>MSWInterrupt</td>
<td>67</td>
<td>Machine software interrupt</td>
</tr>
<tr>
<td>STimerInterrupt</td>
<td>69</td>
<td>Supervisor timer interrupt</td>
</tr>
<tr>
<td>MTimerInterrupt</td>
<td>71</td>
<td>Machine timer interrupt</td>
</tr>
<tr>
<td>SExternalInterrupt</td>
<td>73</td>
<td>Supervisor external interrupt</td>
</tr>
<tr>
<td>MExternalInterrupt</td>
<td>75</td>
<td>Machine external interrupt</td>
</tr>
<tr>
<td>GenericNMI</td>
<td>4294967295</td>
<td>Generic NMI</td>
</tr>
</tbody>
</table>

Table 10.1: Exceptions implemented in: Hart
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: Hart

This level in the model hierarchy has 6 commands.

This level in the model hierarchy has 5 register groups:

<table>
<thead>
<tr>
<th>Group name</th>
<th>Registers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core</td>
<td>33</td>
</tr>
<tr>
<td>User_Control_and_Status</td>
<td>64</td>
</tr>
<tr>
<td>Supervisor_Control_and_Status</td>
<td>13</td>
</tr>
<tr>
<td>Machine_Control_and_Status</td>
<td>199</td>
</tr>
<tr>
<td>Integration_support</td>
<td>25</td>
</tr>
</tbody>
</table>

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: Hart

12.1.1 debugflags

show or modify the processor debug flags

<table>
<thead>
<tr>
<th>Argument</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-get</td>
<td>Boolean</td>
<td>print current processor flags value</td>
</tr>
<tr>
<td>-mask</td>
<td>Boolean</td>
<td>print valid debug flag bits</td>
</tr>
<tr>
<td>-set</td>
<td>Int32</td>
<td>new processor flags (only flags 0x00000006 can be modified)</td>
</tr>
</tbody>
</table>

Table 12.1: debugflags command arguments

12.1.2 dumpTLB

12.1.2.1 Argument description

show TLB contents

12.1.3 getCSRIndex

Return index for a named CSR (or -1 if no matching CSR)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-name</td>
<td>String</td>
<td>CSR name</td>
</tr>
</tbody>
</table>

Table 12.2: getCSRIndex command arguments

12.1.4 isync

specify instruction address range for synchronous execution
12.1.5 itrace

enable or disable instruction tracing

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

Table 12.4: itrace command arguments

12.1.6 listCSRs

12.1.6.1 Argument description

List all CSRs in index order
Chapter 13

Registers

13.1 Level 1: Hart

13.1.1 Core

Registers at level:1, type:Hart group:Core

<table>
<thead>
<tr>
<th>Name</th>
<th>Bits</th>
<th>Initial-Hex</th>
<th>RW</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>zero</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td></td>
</tr>
<tr>
<td>ra</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td></td>
</tr>
<tr>
<td>sp</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>stack pointer</td>
</tr>
<tr>
<td>gp</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td></td>
</tr>
<tr>
<td>tp</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td></td>
</tr>
<tr>
<td>t0</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td></td>
</tr>
<tr>
<td>t1</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td></td>
</tr>
<tr>
<td>t2</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td></td>
</tr>
<tr>
<td>s0</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td></td>
</tr>
<tr>
<td>s1</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td></td>
</tr>
<tr>
<td>a0</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td></td>
</tr>
<tr>
<td>a1</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td></td>
</tr>
<tr>
<td>a2</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td></td>
</tr>
<tr>
<td>a3</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td></td>
</tr>
<tr>
<td>a4</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td></td>
</tr>
<tr>
<td>a5</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td></td>
</tr>
<tr>
<td>a6</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td></td>
</tr>
<tr>
<td>a7</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td></td>
</tr>
<tr>
<td>s2</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td></td>
</tr>
<tr>
<td>s3</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td></td>
</tr>
<tr>
<td>s4</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td></td>
</tr>
<tr>
<td>s5</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td></td>
</tr>
<tr>
<td>s6</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td></td>
</tr>
<tr>
<td>s7</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td></td>
</tr>
<tr>
<td>s8</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td></td>
</tr>
<tr>
<td>s9</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td></td>
</tr>
<tr>
<td>s10</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td></td>
</tr>
<tr>
<td>s11</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td></td>
</tr>
<tr>
<td>t3</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td></td>
</tr>
<tr>
<td>t4</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td></td>
</tr>
<tr>
<td>t5</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td></td>
</tr>
<tr>
<td>t6</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td></td>
</tr>
<tr>
<td>pc</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>program counter</td>
</tr>
</tbody>
</table>
### 13.1.2 User Control and Status

Registers at level 1, type:Hart group:User Control and Status

<table>
<thead>
<tr>
<th>Name</th>
<th>Bits</th>
<th>Initial-Hex</th>
<th>RW</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cycle</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Cycle Counter</td>
</tr>
<tr>
<td>time</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Timer</td>
</tr>
<tr>
<td>instrret</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Instructions Retired</td>
</tr>
<tr>
<td>hpmcounter3</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor Counter 3</td>
</tr>
<tr>
<td>hpmcounter4</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor Counter 4</td>
</tr>
<tr>
<td>hpmcounter5</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor Counter 5</td>
</tr>
<tr>
<td>hpmcounter6</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor Counter 6</td>
</tr>
<tr>
<td>hpmcounter7</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor Counter 7</td>
</tr>
<tr>
<td>hpmcounter8</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor Counter 8</td>
</tr>
<tr>
<td>hpmcounter9</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor Counter 9</td>
</tr>
<tr>
<td>hpmcounter10</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor Counter 10</td>
</tr>
<tr>
<td>hpmcounter11</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor Counter 11</td>
</tr>
<tr>
<td>hpmcounter12</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor Counter 12</td>
</tr>
<tr>
<td>hpmcounter13</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor Counter 13</td>
</tr>
<tr>
<td>hpmcounter14</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor Counter 14</td>
</tr>
<tr>
<td>hpmcounter15</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor Counter 15</td>
</tr>
<tr>
<td>hpmcounter16</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor Counter 16</td>
</tr>
<tr>
<td>hpmcounter17</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor Counter 17</td>
</tr>
<tr>
<td>hpmcounter18</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor Counter 18</td>
</tr>
<tr>
<td>hpmcounter19</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor Counter 19</td>
</tr>
<tr>
<td>hpmcounter20</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor Counter 20</td>
</tr>
<tr>
<td>hpmcounter21</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor Counter 21</td>
</tr>
<tr>
<td>hpmcounter22</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor Counter 22</td>
</tr>
<tr>
<td>hpmcounter23</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor Counter 23</td>
</tr>
<tr>
<td>hpmcounter24</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor Counter 24</td>
</tr>
<tr>
<td>hpmcounter25</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor Counter 25</td>
</tr>
<tr>
<td>hpmcounter26</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor Counter 26</td>
</tr>
<tr>
<td>hpmcounter27</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor Counter 27</td>
</tr>
<tr>
<td>hpmcounter28</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor Counter 28</td>
</tr>
<tr>
<td>hpmcounter29</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor Counter 29</td>
</tr>
<tr>
<td>hpmcounter30</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor Counter 30</td>
</tr>
<tr>
<td>hpmcounter31</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor Counter 31</td>
</tr>
<tr>
<td>cycleh</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Cycle Counter High</td>
</tr>
<tr>
<td>timeh</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Timer High</td>
</tr>
<tr>
<td>instrreth</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Instructions Retired High</td>
</tr>
<tr>
<td>hpmcounterh3</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor High 3</td>
</tr>
<tr>
<td>hpmcounterh4</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor High 4</td>
</tr>
<tr>
<td>hpmcounterh5</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor High 5</td>
</tr>
<tr>
<td>hpmcounterh6</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor High 6</td>
</tr>
<tr>
<td>hpmcounterh7</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor High 7</td>
</tr>
<tr>
<td>hpmcounterh8</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor High 8</td>
</tr>
<tr>
<td>hpmcounterh9</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor High 9</td>
</tr>
<tr>
<td>hpmcounterh10</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor High 10</td>
</tr>
<tr>
<td>hpmcounterh11</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor High 11</td>
</tr>
<tr>
<td>hpmcounterh12</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor High 12</td>
</tr>
<tr>
<td>hpmcounterh13</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor High 13</td>
</tr>
<tr>
<td>hpmcounterh14</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor High 14</td>
</tr>
<tr>
<td>hpmcounterh15</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor High 15</td>
</tr>
<tr>
<td>Name</td>
<td>Bits</td>
<td>Initial-Hex</td>
<td>RW</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>------</td>
<td>-------------</td>
<td>----</td>
<td>--------------------------------------</td>
</tr>
<tr>
<td>hpmcounterh16</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor High 16</td>
</tr>
<tr>
<td>hpmcounterh17</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor High 17</td>
</tr>
<tr>
<td>hpmcounterh18</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor High 18</td>
</tr>
<tr>
<td>hpmcounterh19</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor High 19</td>
</tr>
<tr>
<td>hpmcounterh20</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor High 20</td>
</tr>
<tr>
<td>hpmcounterh21</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor High 21</td>
</tr>
<tr>
<td>hpmcounterh22</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor High 22</td>
</tr>
<tr>
<td>hpmcounterh23</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor High 23</td>
</tr>
<tr>
<td>hpmcounterh24</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor High 24</td>
</tr>
<tr>
<td>hpmcounterh25</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor High 25</td>
</tr>
<tr>
<td>hpmcounterh26</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor High 26</td>
</tr>
<tr>
<td>hpmcounterh27</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor High 27</td>
</tr>
<tr>
<td>hpmcounterh28</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor High 28</td>
</tr>
<tr>
<td>hpmcounterh29</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor High 29</td>
</tr>
<tr>
<td>hpmcounterh30</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor High 30</td>
</tr>
<tr>
<td>hpmcounterh31</td>
<td>32</td>
<td>0</td>
<td>r-</td>
<td>Performance Monitor High 31</td>
</tr>
</tbody>
</table>

Table 13.2: Registers at level 1, type:Hart group:User_Control_and_Status

### 13.1.3 Supervisor_Control_and_Status

Registers at level:1, type:Hart group:Supervisor_Control_and_Status

<table>
<thead>
<tr>
<th>Name</th>
<th>Bits</th>
<th>Initial-Hex</th>
<th>RW</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sstatus</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Supervisor Status</td>
</tr>
<tr>
<td>sie</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Supervisor Interrupt Enable</td>
</tr>
<tr>
<td>stvec</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Supervisor Trap-Vector Base-Address</td>
</tr>
<tr>
<td>scounteren</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Supervisor Counter Enable</td>
</tr>
<tr>
<td>senvcfg</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Supervisor Environment Configuration</td>
</tr>
<tr>
<td>sscratch</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Supervisor Scratch</td>
</tr>
<tr>
<td>sepc</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Supervisor Exception Program Counter</td>
</tr>
<tr>
<td>scause</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Supervisor Cause</td>
</tr>
<tr>
<td>stval</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Supervisor Trap Value</td>
</tr>
<tr>
<td>sip</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Supervisor Interrupt Pending</td>
</tr>
<tr>
<td>satp</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Supervisor Address Translation and Protection</td>
</tr>
<tr>
<td>scontext</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Trigger Supervisor Context</td>
</tr>
<tr>
<td>mscontext</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Trigger Machine Context Alias</td>
</tr>
</tbody>
</table>

Table 13.3: Registers at level 1, type:Hart group:Supervisor_Control_and_Status

### 13.1.4 Machine_Control_and_Status

Registers at level:1, type:Hart group:Machine_Control_and_Status

<table>
<thead>
<tr>
<th>Name</th>
<th>Bits</th>
<th>Initial-Hex</th>
<th>RW</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mstatus</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Status</td>
</tr>
<tr>
<td>misa</td>
<td>32</td>
<td>40141100</td>
<td>rw</td>
<td>ISA and Extensions</td>
</tr>
<tr>
<td>medeleq</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Exception Delegation</td>
</tr>
<tr>
<td>mideleg</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Interrupt Delegation</td>
</tr>
<tr>
<td>mie</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Interrupt Enable</td>
</tr>
<tr>
<td>mtvec</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Trap-Vector Base-Address</td>
</tr>
<tr>
<td>mcouteren</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Counter Enable</td>
</tr>
<tr>
<td>menvcfg</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Environment Configuration</td>
</tr>
<tr>
<td>mstatush</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Status High</td>
</tr>
<tr>
<td>menvcfggh</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Environment Configuration High</td>
</tr>
<tr>
<td>Name</td>
<td>Size</td>
<td>Access</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>------------------</td>
<td>------</td>
<td>--------</td>
<td>--------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>mcountinhibit</td>
<td>32</td>
<td>rw</td>
<td>Machine Counter Inhibit</td>
<td></td>
</tr>
<tr>
<td>mhpmevent3</td>
<td>32</td>
<td>rw</td>
<td>Machine Performance Monitor Event Select 3</td>
<td></td>
</tr>
<tr>
<td>mhpmevent4</td>
<td>32</td>
<td>rw</td>
<td>Machine Performance Monitor Event Select 4</td>
<td></td>
</tr>
<tr>
<td>mhpmevent5</td>
<td>32</td>
<td>rw</td>
<td>Machine Performance Monitor Event Select 5</td>
<td></td>
</tr>
<tr>
<td>mhpmevent6</td>
<td>32</td>
<td>rw</td>
<td>Machine Performance Monitor Event Select 6</td>
<td></td>
</tr>
<tr>
<td>mhpmevent7</td>
<td>32</td>
<td>rw</td>
<td>Machine Performance Monitor Event Select 7</td>
<td></td>
</tr>
<tr>
<td>mhpmevent8</td>
<td>32</td>
<td>rw</td>
<td>Machine Performance Monitor Event Select 8</td>
<td></td>
</tr>
<tr>
<td>mhpmevent9</td>
<td>32</td>
<td>rw</td>
<td>Machine Performance Monitor Event Select 9</td>
<td></td>
</tr>
<tr>
<td>mhpmevent10</td>
<td>32</td>
<td>rw</td>
<td>Machine Performance Monitor Event Select 10</td>
<td></td>
</tr>
<tr>
<td>mhpmevent11</td>
<td>32</td>
<td>rw</td>
<td>Machine Performance Monitor Event Select 11</td>
<td></td>
</tr>
<tr>
<td>mhpmevent12</td>
<td>32</td>
<td>rw</td>
<td>Machine Performance Monitor Event Select 12</td>
<td></td>
</tr>
<tr>
<td>mhpmevent13</td>
<td>32</td>
<td>rw</td>
<td>Machine Performance Monitor Event Select 13</td>
<td></td>
</tr>
<tr>
<td>mhpmevent14</td>
<td>32</td>
<td>rw</td>
<td>Machine Performance Monitor Event Select 14</td>
<td></td>
</tr>
<tr>
<td>mhpmevent15</td>
<td>32</td>
<td>rw</td>
<td>Machine Performance Monitor Event Select 15</td>
<td></td>
</tr>
<tr>
<td>mhpmevent16</td>
<td>32</td>
<td>rw</td>
<td>Machine Performance Monitor Event Select 16</td>
<td></td>
</tr>
<tr>
<td>mhpmevent17</td>
<td>32</td>
<td>rw</td>
<td>Machine Performance Monitor Event Select 17</td>
<td></td>
</tr>
<tr>
<td>mhpmevent18</td>
<td>32</td>
<td>rw</td>
<td>Machine Performance Monitor Event Select 18</td>
<td></td>
</tr>
<tr>
<td>mhpmevent19</td>
<td>32</td>
<td>rw</td>
<td>Machine Performance Monitor Event Select 19</td>
<td></td>
</tr>
<tr>
<td>mhpmevent20</td>
<td>32</td>
<td>rw</td>
<td>Machine Performance Monitor Event Select 20</td>
<td></td>
</tr>
<tr>
<td>mhpmevent21</td>
<td>32</td>
<td>rw</td>
<td>Machine Performance Monitor Event Select 21</td>
<td></td>
</tr>
<tr>
<td>mhpmevent22</td>
<td>32</td>
<td>rw</td>
<td>Machine Performance Monitor Event Select 22</td>
<td></td>
</tr>
<tr>
<td>mhpmevent23</td>
<td>32</td>
<td>rw</td>
<td>Machine Performance Monitor Event Select 23</td>
<td></td>
</tr>
<tr>
<td>mhpmevent24</td>
<td>32</td>
<td>rw</td>
<td>Machine Performance Monitor Event Select 24</td>
<td></td>
</tr>
<tr>
<td>mhpmevent25</td>
<td>32</td>
<td>rw</td>
<td>Machine Performance Monitor Event Select 25</td>
<td></td>
</tr>
<tr>
<td>mhpmevent26</td>
<td>32</td>
<td>rw</td>
<td>Machine Performance Monitor Event Select 26</td>
<td></td>
</tr>
<tr>
<td>mhpmevent27</td>
<td>32</td>
<td>rw</td>
<td>Machine Performance Monitor Event Select 27</td>
<td></td>
</tr>
<tr>
<td>mhpmevent28</td>
<td>32</td>
<td>rw</td>
<td>Machine Performance Monitor Event Select 28</td>
<td></td>
</tr>
<tr>
<td>mhpmevent29</td>
<td>32</td>
<td>rw</td>
<td>Machine Performance Monitor Event Select 29</td>
<td></td>
</tr>
<tr>
<td>mhpmevent30</td>
<td>32</td>
<td>rw</td>
<td>Machine Performance Monitor Event Select 30</td>
<td></td>
</tr>
<tr>
<td>mhpmevent31</td>
<td>32</td>
<td>rw</td>
<td>Machine Performance Monitor Event Select 31</td>
<td></td>
</tr>
<tr>
<td>mscratch</td>
<td>32</td>
<td>rw</td>
<td>Machine Scratch</td>
<td></td>
</tr>
<tr>
<td>mepc</td>
<td>32</td>
<td>rw</td>
<td>Machine Exception Program Counter</td>
<td></td>
</tr>
<tr>
<td>mcause</td>
<td>32</td>
<td>rw</td>
<td>Machine Cause</td>
<td></td>
</tr>
<tr>
<td>mtval</td>
<td>32</td>
<td>rw</td>
<td>Machine Trap Value</td>
<td></td>
</tr>
<tr>
<td>mip</td>
<td>32</td>
<td>rw</td>
<td>Machine Interrupt Pending</td>
<td></td>
</tr>
<tr>
<td>pmpecfg0</td>
<td>32</td>
<td>rw</td>
<td>Physical Memory Protection Configuration 0</td>
<td></td>
</tr>
<tr>
<td>pmpecfg1</td>
<td>32</td>
<td>rw</td>
<td>Physical Memory Protection Configuration 1</td>
<td></td>
</tr>
<tr>
<td>pmpecfg2</td>
<td>32</td>
<td>rw</td>
<td>Physical Memory Protection Configuration 2</td>
<td></td>
</tr>
<tr>
<td>pmpecfg3</td>
<td>32</td>
<td>rw</td>
<td>Physical Memory Protection Configuration 3</td>
<td></td>
</tr>
<tr>
<td>pmpecfg4</td>
<td>32</td>
<td>rw</td>
<td>Physical Memory Protection Configuration 4</td>
<td></td>
</tr>
<tr>
<td>pmpecfg5</td>
<td>32</td>
<td>rw</td>
<td>Physical Memory Protection Configuration 5</td>
<td></td>
</tr>
<tr>
<td>pmpecfg6</td>
<td>32</td>
<td>rw</td>
<td>Physical Memory Protection Configuration 6</td>
<td></td>
</tr>
<tr>
<td>pmpecfg7</td>
<td>32</td>
<td>rw</td>
<td>Physical Memory Protection Configuration 7</td>
<td></td>
</tr>
<tr>
<td>pmpecfg8</td>
<td>32</td>
<td>rw</td>
<td>Physical Memory Protection Configuration 8</td>
<td></td>
</tr>
<tr>
<td>pmpecfg9</td>
<td>32</td>
<td>rw</td>
<td>Physical Memory Protection Configuration 9</td>
<td></td>
</tr>
<tr>
<td>pmpecfg10</td>
<td>32</td>
<td>rw</td>
<td>Physical Memory Protection Configuration 10</td>
<td></td>
</tr>
<tr>
<td>pmpecfg11</td>
<td>32</td>
<td>rw</td>
<td>Physical Memory Protection Configuration 11</td>
<td></td>
</tr>
<tr>
<td>pmpecfg12</td>
<td>32</td>
<td>rw</td>
<td>Physical Memory Protection Configuration 12</td>
<td></td>
</tr>
<tr>
<td>pmpecfg13</td>
<td>32</td>
<td>rw</td>
<td>Physical Memory Protection Configuration 13</td>
<td></td>
</tr>
<tr>
<td>pmpecfg14</td>
<td>32</td>
<td>rw</td>
<td>Physical Memory Protection Configuration 14</td>
<td></td>
</tr>
<tr>
<td>pmpecfg15</td>
<td>32</td>
<td>rw</td>
<td>Physical Memory Protection Configuration 15</td>
<td></td>
</tr>
<tr>
<td>pmppaddr0</td>
<td>32</td>
<td>rw</td>
<td>Physical Memory Protection Address 0</td>
<td></td>
</tr>
<tr>
<td>pmppaddr1</td>
<td>32</td>
<td>rw</td>
<td>Physical Memory Protection Address 1</td>
<td></td>
</tr>
<tr>
<td>pmppaddr2</td>
<td>32</td>
<td>rw</td>
<td>Physical Memory Protection Address 2</td>
<td></td>
</tr>
<tr>
<td>pmppaddr3</td>
<td>32</td>
<td>rw</td>
<td>Physical Memory Protection Address 3</td>
<td></td>
</tr>
<tr>
<td>pmppaddr4</td>
<td>32</td>
<td>rw</td>
<td>Physical Memory Protection Address 4</td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>Size</td>
<td>Value</td>
<td>Mode</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>------</td>
<td>-------</td>
<td>------</td>
<td>-----------------------------------</td>
</tr>
<tr>
<td>pmpaddr5</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 5</td>
</tr>
<tr>
<td>pmpaddr6</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 6</td>
</tr>
<tr>
<td>pmpaddr7</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 7</td>
</tr>
<tr>
<td>pmpaddr8</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 8</td>
</tr>
<tr>
<td>pmpaddr9</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 9</td>
</tr>
<tr>
<td>pmpaddr10</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 10</td>
</tr>
<tr>
<td>pmpaddr11</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 11</td>
</tr>
<tr>
<td>pmpaddr12</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 12</td>
</tr>
<tr>
<td>pmpaddr13</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 13</td>
</tr>
<tr>
<td>pmpaddr14</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 14</td>
</tr>
<tr>
<td>pmpaddr15</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 15</td>
</tr>
<tr>
<td>pmpaddr16</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 16</td>
</tr>
<tr>
<td>pmpaddr17</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 17</td>
</tr>
<tr>
<td>pmpaddr18</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 18</td>
</tr>
<tr>
<td>pmpaddr19</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 19</td>
</tr>
<tr>
<td>pmpaddr20</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 20</td>
</tr>
<tr>
<td>pmpaddr21</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 21</td>
</tr>
<tr>
<td>pmpaddr22</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 22</td>
</tr>
<tr>
<td>pmpaddr23</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 23</td>
</tr>
<tr>
<td>pmpaddr24</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 24</td>
</tr>
<tr>
<td>pmpaddr25</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 25</td>
</tr>
<tr>
<td>pmpaddr26</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 26</td>
</tr>
<tr>
<td>pmpaddr27</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 27</td>
</tr>
<tr>
<td>pmpaddr28</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 28</td>
</tr>
<tr>
<td>pmpaddr29</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 29</td>
</tr>
<tr>
<td>pmpaddr30</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 30</td>
</tr>
<tr>
<td>pmpaddr31</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 31</td>
</tr>
<tr>
<td>pmpaddr32</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 32</td>
</tr>
<tr>
<td>pmpaddr33</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 33</td>
</tr>
<tr>
<td>pmpaddr34</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 34</td>
</tr>
<tr>
<td>pmpaddr35</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 35</td>
</tr>
<tr>
<td>pmpaddr36</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 36</td>
</tr>
<tr>
<td>pmpaddr37</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 37</td>
</tr>
<tr>
<td>pmpaddr38</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 38</td>
</tr>
<tr>
<td>pmpaddr39</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 39</td>
</tr>
<tr>
<td>pmpaddr40</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 40</td>
</tr>
<tr>
<td>pmpaddr41</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 41</td>
</tr>
<tr>
<td>pmpaddr42</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 42</td>
</tr>
<tr>
<td>pmpaddr43</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 43</td>
</tr>
<tr>
<td>pmpaddr44</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 44</td>
</tr>
<tr>
<td>pmpaddr45</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 45</td>
</tr>
<tr>
<td>pmpaddr46</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 46</td>
</tr>
<tr>
<td>pmpaddr47</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 47</td>
</tr>
<tr>
<td>pmpaddr48</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 48</td>
</tr>
<tr>
<td>pmpaddr49</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 49</td>
</tr>
<tr>
<td>pmpaddr50</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 50</td>
</tr>
<tr>
<td>pmpaddr51</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 51</td>
</tr>
<tr>
<td>pmpaddr52</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 52</td>
</tr>
<tr>
<td>pmpaddr53</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 53</td>
</tr>
<tr>
<td>pmpaddr54</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 54</td>
</tr>
<tr>
<td>pmpaddr55</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 55</td>
</tr>
<tr>
<td>pmpaddr56</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 56</td>
</tr>
<tr>
<td>pmpaddr57</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 57</td>
</tr>
<tr>
<td>pmpaddr58</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 58</td>
</tr>
<tr>
<td>pmpaddr59</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 59</td>
</tr>
<tr>
<td>pmpaddr60</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 60</td>
</tr>
<tr>
<td>Symbol</td>
<td>Size</td>
<td>Alignment</td>
<td>Access</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>------</td>
<td>-----------</td>
<td>--------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>pmpaddr61</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 61</td>
</tr>
<tr>
<td>pmpaddr62</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 62</td>
</tr>
<tr>
<td>pmpaddr63</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Physical Memory Protection Address 63</td>
</tr>
<tr>
<td>tselect</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Trigger Register Select</td>
</tr>
<tr>
<td>tdata1</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Trigger Data 1</td>
</tr>
<tr>
<td>tdata2</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Trigger Data 2</td>
</tr>
<tr>
<td>tdata3</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Trigger Data 3</td>
</tr>
<tr>
<td>tinfo</td>
<td>32</td>
<td>7d</td>
<td>rw</td>
<td>Trigger Info</td>
</tr>
<tr>
<td>tcontrol</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Trigger Control</td>
</tr>
<tr>
<td>mcontext</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Trigger Machine Context</td>
</tr>
<tr>
<td>mcycle</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Cycle Counter</td>
</tr>
<tr>
<td>minstret</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Instructions Retired</td>
</tr>
<tr>
<td>mhpmcnt3</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Performance Monitor Counter 3</td>
</tr>
<tr>
<td>mhpmcnt4</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Performance Monitor Counter 4</td>
</tr>
<tr>
<td>mhpmcnt5</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Performance Monitor Counter 5</td>
</tr>
<tr>
<td>mhpmcnt6</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Performance Monitor Counter 6</td>
</tr>
<tr>
<td>mhpmcnt7</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Performance Monitor Counter 7</td>
</tr>
<tr>
<td>mhpmcnt8</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Performance Monitor Counter 8</td>
</tr>
<tr>
<td>mhpmcnt9</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Performance Monitor Counter 9</td>
</tr>
<tr>
<td>mhpmcnt10</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Performance Monitor Counter 10</td>
</tr>
<tr>
<td>mhpmcnt11</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Performance Monitor Counter 11</td>
</tr>
<tr>
<td>mhpmcnt12</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Performance Monitor Counter 12</td>
</tr>
<tr>
<td>mhpmcnt13</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Performance Monitor Counter 13</td>
</tr>
<tr>
<td>mhpmcnt14</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Performance Monitor Counter 14</td>
</tr>
<tr>
<td>mhpmcnt15</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Performance Monitor Counter 15</td>
</tr>
<tr>
<td>mhpmcnt16</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Performance Monitor Counter 16</td>
</tr>
<tr>
<td>mhpmcnt17</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Performance Monitor Counter 17</td>
</tr>
<tr>
<td>mhpmcnt18</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Performance Monitor Counter 18</td>
</tr>
<tr>
<td>mhpmcnt19</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Performance Monitor Counter 19</td>
</tr>
<tr>
<td>mhpmcnt20</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Performance Monitor Counter 20</td>
</tr>
<tr>
<td>mhpmcnt21</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Performance Monitor Counter 21</td>
</tr>
<tr>
<td>mhpmcnt22</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Performance Monitor Counter 22</td>
</tr>
<tr>
<td>mhpmcnt23</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Performance Monitor Counter 23</td>
</tr>
<tr>
<td>mhpmcnt24</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Performance Monitor Counter 24</td>
</tr>
<tr>
<td>mhpmcnt25</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Performance Monitor Counter 25</td>
</tr>
<tr>
<td>mhpmcnt26</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Performance Monitor Counter 26</td>
</tr>
<tr>
<td>mhpmcnt27</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Performance Monitor Counter 27</td>
</tr>
<tr>
<td>mhpmcnt28</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Performance Monitor Counter 28</td>
</tr>
<tr>
<td>mhpmcnt29</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Performance Monitor Counter 29</td>
</tr>
<tr>
<td>mhpmcnt30</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Performance Monitor Counter 30</td>
</tr>
<tr>
<td>mhpmcnt31</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Performance Monitor Counter 31</td>
</tr>
<tr>
<td>mcycleh</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Cycle Counter High</td>
</tr>
<tr>
<td>minstreth</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Instructions Retired High</td>
</tr>
<tr>
<td>mhpmcnth3</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Performance Monitor Counter High 3</td>
</tr>
<tr>
<td>mhpmcnth4</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Performance Monitor Counter High 4</td>
</tr>
<tr>
<td>mhpmcnth5</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Performance Monitor Counter High 5</td>
</tr>
<tr>
<td>mhpmcnth6</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Performance Monitor Counter High 6</td>
</tr>
<tr>
<td>mhpmcnth7</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Performance Monitor Counter High 7</td>
</tr>
<tr>
<td>mhpmcnth8</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Performance Monitor Counter High 8</td>
</tr>
<tr>
<td>mhpmcnth9</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Performance Monitor Counter High 9</td>
</tr>
<tr>
<td>mhpmcnth10</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Performance Monitor Counter High 10</td>
</tr>
<tr>
<td>mhpmcnth11</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Performance Monitor Counter High 11</td>
</tr>
<tr>
<td>mhpmcnth12</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Performance Monitor Counter High 12</td>
</tr>
<tr>
<td>mhpmcnth13</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Performance Monitor Counter High 13</td>
</tr>
<tr>
<td>mhpmcnth14</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Performance Monitor Counter High 14</td>
</tr>
<tr>
<td>mhpmcnth15</td>
<td>32</td>
<td>0</td>
<td>rw</td>
<td>Machine Performance Monitor Counter High 15</td>
</tr>
</tbody>
</table>
### 13.1.5 Integration_support

Registers at level:1, type:Hart group:Integration_support

<table>
<thead>
<tr>
<th>Name</th>
<th>Bits</th>
<th>Initial-Hex</th>
<th>RW</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>commercial</td>
<td>8</td>
<td>0</td>
<td>r-</td>
<td>Commercial feature in use</td>
</tr>
<tr>
<td>PTWStage</td>
<td>8</td>
<td>0</td>
<td>r-</td>
<td>PTW active stage (0:none 1:HS 2:VS 3:G)</td>
</tr>
<tr>
<td>PTWInputAddr</td>
<td>64</td>
<td>0</td>
<td>r-</td>
<td>PTW input address</td>
</tr>
<tr>
<td>PTWLevel</td>
<td>8</td>
<td>0</td>
<td>r-</td>
<td>PTW active level</td>
</tr>
<tr>
<td>ASYNCEPE</td>
<td>8</td>
<td>0</td>
<td>r-</td>
<td>Asynchronous Event Pending &amp; Enabled</td>
</tr>
<tr>
<td>mask_pmpcfg0</td>
<td>64</td>
<td>ffffffff ffffffff</td>
<td>r-</td>
<td>Write mask for pmpcfg0</td>
</tr>
<tr>
<td>mask_pmpcfg1</td>
<td>32</td>
<td>ffffffff</td>
<td>r-</td>
<td>Write mask for pmpcfg1</td>
</tr>
<tr>
<td>mask_pmpcfg2</td>
<td>32</td>
<td>ffffffff</td>
<td>r-</td>
<td>Write mask for pmpcfg2</td>
</tr>
<tr>
<td>mask_pmpcfg3</td>
<td>32</td>
<td>ffffffff</td>
<td>r-</td>
<td>Write mask for pmpcfg3</td>
</tr>
<tr>
<td>mask_pmpaddr0</td>
<td>64</td>
<td>ffffffff ffffffff</td>
<td>r-</td>
<td>Write mask for pmpaddr0</td>
</tr>
<tr>
<td>mask_pmpaddr1</td>
<td>64</td>
<td>ffffffff ffffffff</td>
<td>r-</td>
<td>Write mask for pmpaddr1</td>
</tr>
<tr>
<td>mask_pmpaddr2</td>
<td>64</td>
<td>ffffffff ffffffff</td>
<td>r-</td>
<td>Write mask for pmpaddr2</td>
</tr>
<tr>
<td>mask_pmpaddr3</td>
<td>64</td>
<td>ffffffff ffffffff</td>
<td>r-</td>
<td>Write mask for pmpaddr3</td>
</tr>
<tr>
<td>mask_pmpaddr4</td>
<td>64</td>
<td>ffffffff ffffffff</td>
<td>r-</td>
<td>Write mask for pmpaddr4</td>
</tr>
<tr>
<td>mask_pmpaddr5</td>
<td>64</td>
<td>ffffffff ffffffff</td>
<td>r-</td>
<td>Write mask for pmpaddr5</td>
</tr>
<tr>
<td>mask_pmpaddr6</td>
<td>64</td>
<td>ffffffff ffffffff</td>
<td>r-</td>
<td>Write mask for pmpaddr6</td>
</tr>
<tr>
<td>mask_pmpaddr7</td>
<td>64</td>
<td>ffffffff ffffffff</td>
<td>r-</td>
<td>Write mask for pmpaddr7</td>
</tr>
<tr>
<td>mask_pmpaddr8</td>
<td>64</td>
<td>ffffffff ffffffff</td>
<td>r-</td>
<td>Write mask for pmpaddr8</td>
</tr>
<tr>
<td>mask_pmpaddr9</td>
<td>64</td>
<td>ffffffff ffffffff</td>
<td>r-</td>
<td>Write mask for pmpaddr9</td>
</tr>
<tr>
<td>mask_pmpaddr10</td>
<td>64</td>
<td>ffffffff ffffffff</td>
<td>r-</td>
<td>Write mask for pmpaddr10</td>
</tr>
<tr>
<td>mask_pmpaddr11</td>
<td>64</td>
<td>ffffffff ffffffff</td>
<td>r-</td>
<td>Write mask for pmpaddr11</td>
</tr>
<tr>
<td>mask_pmpaddr12</td>
<td>64</td>
<td>ffffffff ffffffff</td>
<td>r-</td>
<td>Write mask for pmpaddr12</td>
</tr>
<tr>
<td>mask_pmpaddr13</td>
<td>64</td>
<td>ffffffff ffffffff</td>
<td>r-</td>
<td>Write mask for pmpaddr13</td>
</tr>
<tr>
<td>mask_pmpaddr14</td>
<td>64</td>
<td>ffffffff ffffffff</td>
<td>r-</td>
<td>Write mask for pmpaddr14</td>
</tr>
<tr>
<td>mask_pmpaddr15</td>
<td>64</td>
<td>ffffffff ffffffff</td>
<td>r-</td>
<td>Write mask for pmpaddr15</td>
</tr>
</tbody>
</table>

Table 13.5: Registers at level 1, type:Hart group:Integration_support