



OVP Guide to Using Processor Models

Model Specific Information for variant ARM_Cortex-A9MPx3

Imperas Software Limited

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



| | |
|----------|--|
| Author | Imperas Software Limited |
| Version | 0.5 |
| Filename | OVP_Model_Specific_Information_arm_Cortex-A9MPx3.pdf |
| Created | 27 February 2018 |
| Status | OVP Standard Release |

Copyright Notice

All rights reserved. This software and documentation contain information that is the property of Imperas Software Limited. The software and documentation are furnished under a license agreement and may be used or copied only in accordance with the terms of the license agreement. No part of the software and documentation may be reproduced, transmitted, or translated, in any form or by any means, electronic, mechanical, manual, optical, or otherwise, without prior written permission of Imperas Software Limited, or as expressly provided by the license agreement.

Right to Copy Documentation

The license agreement with Imperas permits licensee to make copies of the documentation for its internal use only. Each copy shall include all copyrights, trademarks, service marks, and proprietary rights notices, if any.

Destination Control Statement

All technical data contained in this publication is subject to the export control laws of the United States of America. Disclosure to nationals of other countries contrary to United States law is prohibited. It is the reader's responsibility to determine the applicable regulations and to comply with them.

Disclaimer

IMPERAS SOFTWARE LIMITED., AND ITS LICENSORS MAKE NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

Model Release Status

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

Table of Contents

| | |
|--|----|
| 1 Overview..... | 5 |
| 1.1 Description..... | 5 |
| 1.2 Licensing..... | 5 |
| 1.3 Limitations..... | 5 |
| 1.4 Verification..... | 6 |
| 1.5 Features..... | 6 |
| 1.5.1 Core Features..... | 6 |
| 1.5.2 Memory System..... | 6 |
| 1.5.3 Advanced SIMD and Floating-Point Features..... | 6 |
| 1.5.4 Generic Interrupt Controller..... | 6 |
| 1.6 Debug Mask..... | 6 |
| 1.7 AArch32 Unpredictable Behavior..... | 7 |
| 1.7.1 Equal Target Registers..... | 7 |
| 1.7.2 Floating Point Load/Store Multiple Lists..... | 7 |
| 1.7.3 Floating Point VLD[2-4]/VST[2-4] Range Overflow..... | 7 |
| 1.7.4 If-Then (IT) Block Constraints..... | 7 |
| 1.7.5 Use of R13..... | 7 |
| 1.7.6 Use of R15..... | 7 |
| 1.8 Integration Support..... | 8 |
| 1.8.1 Memory Transaction Query..... | 8 |
| 1.8.2 Page Table Walk Query..... | 8 |
| 1.8.3 Artifact Page Table Walks..... | 8 |
| 1.8.4 MMU and Page Table Walk Events..... | 8 |
| 1.8.5 Artifact Address Translations..... | 8 |
| 1.8.6 Halt Reason Introspection..... | 9 |
| 1.8.7 System Register Access Monitor..... | 9 |
| 1.8.8 System Register Implementation..... | 9 |
| 2 Configuration..... | 9 |
| 2.1 Location..... | 9 |
| 2.2 GDB Path..... | 9 |
| 2.3 Semi-Host Library..... | 9 |
| 2.4 Processor Endian-ness..... | 9 |
| 2.5 QuantumLeap Support..... | 9 |
| 2.6 Processor ELF Code..... | 9 |
| 3 Other Variants in this Model..... | 9 |
| 4 Bus Ports..... | 12 |
| 5 Net Ports..... | 12 |
| 6 FIFO Ports..... | 15 |
| 7 Parameters..... | 15 |
| 8 Execution Modes..... | 19 |
| 9 Exceptions..... | 19 |
| 10 Hierarchy of the model..... | 21 |

| | |
|--|----|
| 10.1 Level 1: MPCORE..... | 21 |
| 10.2 Level 2: CPU..... | 21 |
| 11 Model Commands..... | 23 |
| 11.1 Level 1: MPCORE..... | 23 |
| 11.1.1 isync..... | 23 |
| 11.1.2 itrace..... | 23 |
| 11.2 Level 2: CPU..... | 23 |
| 11.2.1 debugflags..... | 23 |
| 11.2.2 dumpTLB..... | 23 |
| 11.2.3 isync..... | 23 |
| 11.2.4 itrace..... | 24 |
| 11.2.5 validateTLB..... | 24 |
| 12 Registers..... | 24 |
| 12.1 Level 1: MPCORE..... | 24 |
| 12.2 Level 2: CPU..... | 24 |
| 12.2.1 Core..... | 24 |
| 12.2.2 Control..... | 25 |
| 12.2.3 User..... | 25 |
| 12.2.4 FIQ..... | 25 |
| 12.2.5 IRQ..... | 25 |
| 12.2.6 Supervisor..... | 26 |
| 12.2.7 Monitor..... | 26 |
| 12.2.8 Undefined..... | 26 |
| 12.2.9 Abort..... | 26 |
| 12.2.10 SIMD_VFP..... | 26 |
| 12.2.11 SIMD_VFP_SYS..... | 27 |
| 12.2.12 Coprocessor_32_bit..... | 28 |
| 12.2.13 Coprocessor_32_bit_secure..... | 31 |
| 12.2.14 Coprocessor_32_bit_non_secure..... | 31 |
| 12.2.15 Integration_support..... | 32 |
| 12.2.16 MPCore_SCR..... | 33 |
| 12.2.17 MPCore_distributor..... | 33 |
| 12.2.18 MPCore_processor_interface..... | 35 |
| 12.2.19 MPCore_timer_and_watchdog..... | 36 |
| 12.2.20 MPCore_global_timer..... | 36 |

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

ARM Processor Model

1.2 Licensing

Usage of binary model under license governing simulator usage.

Note that for models of ARM CPUs the license includes the following terms:

Licensee is granted a non-exclusive, worldwide, non-transferable, revocable licence to:

If no source is being provided to the Licensee: use and copy only (no modifications rights are granted) the model for the sole purpose of designing, developing, analyzing, debugging, testing, verifying, validating and optimizing software which: (a) (i) is for ARM based systems; and (ii) does not incorporate the ARM Models or any part thereof; and (b) such ARM Models may not be used to emulate an ARM based system to run application software in a production or live environment.

If source code is being provided to the Licensee: use, copy and modify the model for the sole purpose of designing, developing, analyzing, debugging, testing, verifying, validating and optimizing software which: (a) (i) is for ARM based systems; and (ii) does not incorporate the ARM Models or any part thereof; and (b) such ARM Models may not be used to emulate an ARM based system to run application software in a production or live environment.

In the case of any Licensee who is either or both an academic or educational institution the purposes shall be limited to internal use.

Except to the extent that such activity is permitted by applicable law, Licensee shall not reverse engineer, decompile, or disassemble this model. If this model was provided to Licensee in Europe, Licensee shall not reverse engineer, decompile or disassemble the Model for the purposes of error correction.

The License agreement does not entitle Licensee to manufacture in silicon any product based on this model.

The License agreement does not entitle Licensee to use this model for evaluating the validity of any ARM patent.

Source of model available under separate Imperas Software License Agreement.

1.3 Limitations

Instruction pipelines are not modeled in any way. All instructions are assumed to complete immediately. This means that instruction barrier instructions (e.g. ISB, CP15ISB) are treated as NOPs, with the exception of any undefined instruction behavior, which is modeled. The model does not implement speculative fetch behavior. The branch cache is not 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 (as if the memory was of Strongly Ordered or Device-nGnRnE type). Data barrier instructions (e.g. DSB, CP15DSB) are treated as NOPs, with the exception of any undefined instruction behavior, which is modeled. Cache manipulation instructions are implemented as NOPs, with the exception of any undefined instruction behavior, which is modeled.

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

Performance Monitors are implemented as a register interface only except for the cycle counter, which is implemented assuming one instruction per cycle.

TLBs are 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.

1.4 Verification

Models have been extensively tested by Imperas. ARM Cortex-A models have been successfully used by customers to simulate SMP Linux, Ubuntu Desktop, VxWorks and ThreadX on Xilinx Zynq virtual platforms.

1.5 Features

1.5.1 Core Features

Thumb-2 instructions are supported.

Trivial Jazelle extension is implemented.

1.5.2 Memory System

Security extensions are implemented (also known as TrustZone). Non-secure accesses can be made visible externally by connecting the processor to a 41-bit physical bus, in which case bits 39..0 give the true physical address and bit 40 is the NS bit.

VMSA secure and non-secure address translation is implemented.

1.5.3 Advanced SIMD and Floating-Point Features

SIMD and VFP instructions are implemented.

The model implements trapped exceptions if FPTrap is set to 1 in MVFR0 (for AArch32) or MVFR0_EL1 (for AArch64). When floating point exception traps are taken, cumulative exception flags are not updated (in other words, cumulative flag state is always the same as prior to instruction execution, even for SIMD instructions). When multiple enabled exceptions are raised by a single floating point operation, the exception reported is the one in least-significant bit position in FPSCR (for AArch32) or FPCR (for AArch64).

When multiple enabled exceptions are raised by different SIMD element computations, the exception reported is selected from the lowest-index-number SIMD operation. Contact Imperas if requirements for exception reporting differ from these.

Trapped exceptions not are implemented in this variant (FPTrap=0)

1.5.4 Generic Interrupt Controller

GIC block is implemented (GICv1, including security extensions). Accesses to GIC registers can be viewed externally by connecting to the 32-bit GICRegisters bus port. Secure register accesses are at offset 0x0 on this bus; for example, a secure access to GIC register ICDDCR can be observed by monitoring address 0x00001000. Non-secure accesses are at offset 0x80000000 on this bus; for example, a non-secure access to GIC register ICDDCR can be observed by monitoring address 0x80001000

1.6 Debug Mask

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

Value 0x004: enable debugging of MMU/MPU mappings

Value 0x020: enable debugging of reads and writes of GIC block registers.

Value 0x040: enable debugging of exception routing via the GIC model component.

Value 0x080: enable debugging of all system register accesses.

Value 0x100: enable debugging of all traps of system register accesses.

Value 0x200: enable verbose debugging of other miscellaneous behavior (for example, the reason why a particular instruction is undefined).

Value 0x400: enable debugging of Performance Monitor timers

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

1.7 AArch32 Unpredictable Behavior

Many AArch32 instruction behaviors are described in the ARM ARM as CONSTRAINED UNPREDICTABLE. This section describes how such situations are handled by this model.

1.7.1 Equal Target Registers

Some instructions allow the specification of two target registers (for example, double-width SMULL, or some VMOV variants), and such instructions are CONSTRAINED UNPREDICTABLE if the same target register is specified in both positions. In this model, such instructions are treated as UNDEFINED.

1.7.2 Floating Point Load/Store Multiple Lists

Instructions that load or store a list of floating point registers (e.g. VSTM, VLDM, VPUSH, VPOP) are CONSTRAINED UNPREDICTABLE if either the uppermost register in the specified range is greater than 32 or (for 64-bit registers) if more than 16 registers are specified. In this model, such instructions are treated as UNDEFINED.

1.7.3 Floating Point VLD[2-4]/VST[2-4] Range Overflow

Instructions that load or store a fixed number of floating point registers (e.g. VST2, VLD2) are CONSTRAINED UNPREDICTABLE if the upper register bound exceeds the number of implemented floating point registers. In this model, these instructions load and store using modulo 32 indexing (consistent with AArch64 instructions with similar behavior).

1.7.4 If-Then (IT) Block Constraints

Where the behavior of an instruction in an if-then (IT) block is described as CONSTRAINED UNPREDICTABLE, this model treats that instruction as UNDEFINED.

1.7.5 Use of R13

In architecture variants before ARMv8, use of R13 was described as CONSTRAINED UNPREDICTABLE in many circumstances. From ARMv8, most of these situations are no longer considered unpredictable. This model allows R13 to be used like any other GPR, consistent with the ARMv8 specification.

1.7.6 Use of R15

Use of R15 is described as CONSTRAINED UNPREDICTABLE in many circumstances.

This model allows such use to be configured using the parameter "unpredictable" as follows:

Value "undefined": any reference to R15 in such a situation is treated as UNDEFINED;

Value "nop": any reference to R15 in such a situation causes the instruction to be treated as a NOP;

Value "raz_wi": any reference to R15 in such a situation causes the instruction to be treated as a RAZ/WI (that is, R15 is read as zero and write-ignored);

Value "execute": any reference to R15 in such a situation is executed using the current value of R15 on read, and writes to R15 are allowed (but are not interworking).

Value "assert": any reference to R15 in such a situation causes the simulation to halt with an assertion message (allowing any such unpredictable uses to be easily identified).

In this variant, the default is "undefined".

1.8 Integration Support

This model implements a number of non-architectural pseudo-registers and other features to facilitate integration.

1.8.1 Memory Transaction Query

Two registers are intended for use within memory callback functions to provide additional information about the current memory access. Register `transactPL` indicates the processor execution level of the current access (0-3). Note that for load/store translate instructions (e.g. `LDRT`, `STRT`) the reported execution level will be 0, indicating an EL0 access. Register `transactAT` indicates the type of memory access: 0 for a normal read or write; and 1 for a physical access resulting from a page table walk.

1.8.2 Page Table Walk Query

A banked set of registers provides information about the most recently completed page table walk. There are up to six banks of registers: bank 0 is for stage 1 walks, bank 1 is for stage 2 walks, and banks 2-5 are for stage 2 walks initiated by stage 1 level 0-3 entry lookups, respectively. Banks 1-5 are present only for processors with virtualization extensions. The currently active bank can be set using register `PTWBankSelect`. Register `PTWBankValid` is a bitmask indicating which banks contain valid data: for example, the value `0xb` indicates that banks 0, 1 and 3 contain valid data.

Within each bank, there are registers that record addresses and values read during that page table walk. Register `PTWBase` records the table base address. Registers `PTWAddressL0`-`PTWAddressL3` record the addresses of level 0 to level 3 entries read, respectively, and register `PTWAddressValid` is a bitmask indicating which address registers contain valid data: for example, the value `0xe` indicates that `PTWAddressL1`-`PTWAddressL3` are valid but `PTWAddressL0` is not. Registers `PTWValueL0`-`PTWValueL3` contain entry values read at level 0 to level 3. Register `PTWInput` contains the input address that starts a walk and Register `PTWOutput` contains the result address (valid only if the page table walk completes). Register `PTWValueValid` is a bitmask indicating which value registers contain valid data: bits 0-3 indicate `PTWValueL0`-`PTWValueL3`, respectively, bit 4 indicates `PTWBase`, bit 5 indicates `PTWInput` and bit 6 indicates `PTWOutput`.

1.8.3 Artifact Page Table Walks

Registers are also available to enable a simulation environment to initiate an artifact page table walk (for example, to determine the ultimate PA corresponding to a given VA). Register `PTWI_EL1S` initiates a secure EL1 table walk for a fetch. Register `PTWD_EL1S` initiates a secure EL1 table walk for a load or store (note that current ARM processors have unified TLBs, so these registers are synonymous). Registers `PTW[ID]_EL1NS` initiate walks for non-secure EL1 accesses. Registers `PTW[ID]_EL2` initiate EL2 walks. Registers `PTW[ID]_S2` initiate stage 2 walks. Registers `PTW[ID]_EL3` initiate AArch64 EL3 walks. Finally, registers `PTW[ID]_current` initiate current-mode walks (useful in a memory callback context). Each walk fills the query registers described above.

1.8.4 MMU and Page Table Walk Events

Two events are available that allow a simulation environment to be notified on MMU and page table walk actions. Event `mmuEnable` triggers when any MMU is enabled or disabled. Event `pageTableWalk` triggers on completion of any page table walk (including artifact walks).

1.8.5 Artifact Address Translations

A simulation environment can trigger an artifact address translation operation by writing to the architectural address translation registers (e.g. `ATS1CPR`). The results of such translations are written to an integration support register `artifactPAR`, instead of the architectural `PAR` register. This means that such artifact writes will not perturb architectural state.

1.8.6 Halt Reason Introspection

An artifact register HaltReason can be read to determine the reason or reasons that a processor is halted. This register is a bitfield, with the following encoding: bit 0 indicates the processor has executed a wait-for-event (WFE) instruction; bit 1 indicates the processor has executed a wait-for-interrupt (WFI) instruction; and bit 2 indicates the processor is held in reset.

1.8.7 System Register Access Monitor

If parameter "enableSystemMonitorBus" is True, an artifact 32-bit bus "SystemMonitor" is enabled for each PE. Every system register read or write by that PE is then visible as a read or write on this artifact bus, and can therefore be monitored using callbacks installed in the client environment (use opBusReadMonitorAdd/opBusWriteMonitorAdd or icmAddBusReadCallback/icmAddBusWriteCallback, depending on the client API). The format of the address on the bus is as follows:

bits 31:26 - zero

bit 25 - 1 if AArch64 access, 0 if AArch32 access

bit 24 - 1 if non-secure access, 0 if secure access

bits 23:20 - CRm value

bits 19:16 - CRn value

bits 15:12 - op2 value

bits 11:8 - op1 value

bits 7:4 - op0 value (AArch64) or coprocessor number (AArch32)

bits 3:0 - zero

As an example, to view non-secure writes to writes to CNTFRQ_EL0 in AArch64 state, install a write monitor on address range 0x020e0330:0x020e0333.

1.8.8 System Register Implementation

If parameter "enableSystemBus" is True, an artifact 32-bit bus "System" is enabled for each PE. Slave callbacks installed on this bus can be used to implement modified system register behavior (use opBusSlaveNew or icmMapExternalMemory, depending on the client API). The format of the address on the bus is the same as for the system monitor bus, described above.

2 Configuration

2.1 Location

The model source and object file is found in the VLNV tree at:

arm.ovpworld.org/processor/arm/1.0

2.2 GDB Path

The default GDB for this model is found at:

`$IMPERAS_HOME/lib/$IMPERAS_ARCH/gdb/arm-none-eabi-gdb`

2.3 Semi-Host Library

The default semi-host library file is found in the VLNV tree at :

arm.ovpworld.org/semihosting/armNewlib/1.0

2.4 Processor Endian-ness

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

2.5 QuantumLeap Support

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

2.6 Processor ELF Code

The ELF code supported by this model is: 0x28

3 Other Variants in this Model

Table 1. All variants in this model

| Variant |
|----------------|
| ARMv4T |
| ARMv4xM |
| ARMv4 |
| ARMv4TxM |
| ARMv5xM |
| ARMv5 |
| ARMv5TxM |
| ARMv5T |
| ARMv5TExP |
| ARMv5TE |
| ARMv5TEJ |
| ARMv6 |
| ARMv6K |
| ARMv6T2 |
| ARMv6KZ |
| ARMv7 |
| ARM7TDMI |
| ARM7EJ-S |
| ARM720T |
| ARM920T |
| ARM922T |
| ARM926EJ-S |
| ARM940T |
| ARM946E |
| ARM966E |
| ARM968E-S |
| ARM1020E |
| ARM1022E |
| ARM1026EJ-S |
| ARM1136J-S |
| ARM1156T2-S |
| ARM1176JZ-S |
| Cortex-R4 |
| Cortex-R4F |
| Cortex-A5UP |
| Cortex-A5MPx1 |
| Cortex-A5MPx2 |
| Cortex-A5MPx3 |
| Cortex-A5MPx4 |
| Cortex-A8 |

| |
|----------------|
| Cortex-A9UP |
| Cortex-A9MPx1 |
| Cortex-A9MPx2 |
| Cortex-A9MPx3 |
| Cortex-A9MPx4 |
| Cortex-A7UP |
| Cortex-A7MPx1 |
| Cortex-A7MPx2 |
| Cortex-A7MPx3 |
| Cortex-A7MPx4 |
| Cortex-A15UP |
| Cortex-A15MPx1 |
| Cortex-A15MPx2 |
| Cortex-A15MPx3 |
| Cortex-A15MPx4 |
| Cortex-A17MPx1 |
| Cortex-A17MPx2 |
| Cortex-A17MPx3 |
| Cortex-A17MPx4 |
| AArch32 |
| AArch64 |
| Cortex-A32MPx1 |
| Cortex-A32MPx2 |
| Cortex-A32MPx3 |
| Cortex-A32MPx4 |
| Cortex-A35MPx1 |
| Cortex-A35MPx2 |
| Cortex-A35MPx3 |
| Cortex-A35MPx4 |
| Cortex-A53MPx1 |
| Cortex-A53MPx2 |
| Cortex-A53MPx3 |
| Cortex-A53MPx4 |
| Cortex-A55MPx1 |
| Cortex-A55MPx2 |
| Cortex-A55MPx3 |
| Cortex-A55MPx4 |
| Cortex-A57MPx1 |
| Cortex-A57MPx2 |
| Cortex-A57MPx3 |
| Cortex-A57MPx4 |
| Cortex-A72MPx1 |

| |
|----------------|
| Cortex-A72MPx2 |
| Cortex-A72MPx3 |
| Cortex-A72MPx4 |
| Cortex-A73MPx1 |
| Cortex-A73MPx2 |
| Cortex-A73MPx3 |
| Cortex-A73MPx4 |
| Cortex-A75MPx1 |
| Cortex-A75MPx2 |
| Cortex-A75MPx3 |
| Cortex-A75MPx4 |
| MultiCluster |

4 Bus Ports

Table 2. Bus Ports

| Type | Name | min | max | Description |
|--------------------|--------------|-----|-----|----------------------------------|
| master (initiator) | INSTRUCTION | 32 | 41 | |
| master (initiator) | DATA | 32 | 41 | |
| master (initiator) | GICRegisters | 32 | 32 | GIC memory-mapped register block |

5 Net Ports

Table 3. Net Ports

| Name | Type | Description |
|-------|-------|-----------------------------|
| SPI32 | input | Shared peripheral interrupt |
| SPI33 | input | Shared peripheral interrupt |
| SPI34 | input | Shared peripheral interrupt |
| SPI35 | input | Shared peripheral interrupt |
| SPI36 | input | Shared peripheral interrupt |
| SPI37 | input | Shared peripheral interrupt |
| SPI38 | input | Shared peripheral interrupt |
| SPI39 | input | Shared peripheral interrupt |
| SPI40 | input | Shared peripheral interrupt |
| SPI41 | input | Shared peripheral interrupt |
| SPI42 | input | Shared peripheral interrupt |
| SPI43 | input | Shared peripheral interrupt |
| SPI44 | input | Shared peripheral interrupt |
| SPI45 | input | Shared peripheral interrupt |
| SPI46 | input | Shared peripheral interrupt |
| SPI47 | input | Shared peripheral interrupt |
| SPI48 | input | Shared peripheral interrupt |
| SPI49 | input | Shared peripheral interrupt |

| | | |
|-------|-------|-----------------------------|
| SPI50 | input | Shared peripheral interrupt |
| SPI51 | input | Shared peripheral interrupt |
| SPI52 | input | Shared peripheral interrupt |
| SPI53 | input | Shared peripheral interrupt |
| SPI54 | input | Shared peripheral interrupt |
| SPI55 | input | Shared peripheral interrupt |
| SPI56 | input | Shared peripheral interrupt |
| SPI57 | input | Shared peripheral interrupt |
| SPI58 | input | Shared peripheral interrupt |
| SPI59 | input | Shared peripheral interrupt |
| SPI60 | input | Shared peripheral interrupt |
| SPI61 | input | Shared peripheral interrupt |
| SPI62 | input | Shared peripheral interrupt |
| SPI63 | input | Shared peripheral interrupt |
| SPI64 | input | Shared peripheral interrupt |
| SPI65 | input | Shared peripheral interrupt |
| SPI66 | input | Shared peripheral interrupt |
| SPI67 | input | Shared peripheral interrupt |
| SPI68 | input | Shared peripheral interrupt |
| SPI69 | input | Shared peripheral interrupt |
| SPI70 | input | Shared peripheral interrupt |
| SPI71 | input | Shared peripheral interrupt |
| SPI72 | input | Shared peripheral interrupt |
| SPI73 | input | Shared peripheral interrupt |
| SPI74 | input | Shared peripheral interrupt |
| SPI75 | input | Shared peripheral interrupt |
| SPI76 | input | Shared peripheral interrupt |
| SPI77 | input | Shared peripheral interrupt |
| SPI78 | input | Shared peripheral interrupt |
| SPI79 | input | Shared peripheral interrupt |
| SPI80 | input | Shared peripheral interrupt |
| SPI81 | input | Shared peripheral interrupt |
| SPI82 | input | Shared peripheral interrupt |
| SPI83 | input | Shared peripheral interrupt |
| SPI84 | input | Shared peripheral interrupt |
| SPI85 | input | Shared peripheral interrupt |
| SPI86 | input | Shared peripheral interrupt |
| SPI87 | input | Shared peripheral interrupt |
| SPI88 | input | Shared peripheral interrupt |
| SPI89 | input | Shared peripheral interrupt |
| SPI90 | input | Shared peripheral interrupt |
| SPI91 | input | Shared peripheral interrupt |

| | | |
|-------------------|--------|---|
| SPI92 | input | Shared peripheral interrupt |
| SPI93 | input | Shared peripheral interrupt |
| SPI94 | input | Shared peripheral interrupt |
| SPI95 | input | Shared peripheral interrupt |
| SPIVector | input | Shared peripheral interrupt vectorized input |
| periphReset | input | Peripheral reset (active high) |
| CFGSDISABLE | input | Secure configuration lockdown (active high) |
| EVENTI | input | Event input signal, active on rising edge |
| EVENTO | output | Event output signal, active on rising edge |
| wdResetReq_CPU0 | output | Watchdog interrupt request |
| wdReset_CPU0 | input | Watchdog reset (active high) |
| scuReset | input | SCU reset (active high) |
| CLUSTERIDAFF1 | input | Configure MPIDR.Aff1 |
| CLUSTERIDAFF2 | input | Configure MPIDR.Aff2 |
| VINITHI_CPU0 | input | Configure HIVECS mode (SCTLR.V) |
| CFGEND_CPU0 | input | Configure exception endianness (SCTLR.EE) |
| TEINIT_CPU0 | input | Configure exception state at reset (SCTLR.TE) |
| CFGNMFI_CPU0 | input | Configure non-maskable fast interrupts (SCTLR.NMFI) |
| reset_CPU0 | input | Processor reset, active high |
| fiq_CPU0 | input | FIQ interrupt, active high (negation of nFIQ) |
| irq_CPU0 | input | IRQ interrupt, active high (negation of nIRQ) |
| sei_CPU0 | input | System error interrupt, active high |
| AXI_SLVERR_CPU0 | input | AXI external abort type (DECERR=0, SLVERR=1) |
| CP15SDISABLE_CPU0 | input | CP15SDISABLE (active high) |
| wdResetReq_CPU1 | output | Watchdog interrupt request |
| wdReset_CPU1 | input | Watchdog reset (active high) |
| VINITHI_CPU1 | input | Configure HIVECS mode (SCTLR.V) |
| CFGEND_CPU1 | input | Configure exception endianness (SCTLR.EE) |
| TEINIT_CPU1 | input | Configure exception state at reset (SCTLR.TE) |
| CFGNMFI_CPU1 | input | Configure non-maskable fast interrupts (SCTLR.NMFI) |
| reset_CPU1 | input | Processor reset, active high |
| fiq_CPU1 | input | FIQ interrupt, active high (negation of nFIQ) |
| irq_CPU1 | input | IRQ interrupt, active high (negation of nIRQ) |
| sei_CPU1 | input | System error interrupt, active high |
| AXI_SLVERR_CPU1 | input | AXI external abort type (DECERR=0, SLVERR=1) |
| CP15SDISABLE_CPU1 | input | CP15SDISABLE (active high) |
| wdResetReq_CPU2 | output | Watchdog interrupt request |
| wdReset_CPU2 | input | Watchdog reset (active high) |
| VINITHI_CPU2 | input | Configure HIVECS mode (SCTLR.V) |
| CFGEND_CPU2 | input | Configure exception endianness (SCTLR.EE) |

| | | |
|-------------------|-------|---|
| TEINIT_CPU2 | input | Configure exception state at reset (SCTLR.TE) |
| CFGNMFI_CPU2 | input | Configure non-maskable fast interrupts (SCTLR.NMFI) |
| reset_CPU2 | input | Processor reset, active high |
| fiq_CPU2 | input | FIQ interrupt, active high (negation of nFIQ) |
| irq_CPU2 | input | IRQ interrupt, active high (negation of nIRQ) |
| sei_CPU2 | input | System error interrupt, active high |
| AXI_SLVERR_CPU2 | input | AXI external abort type (DECERR=0, SLVERR=1) |
| CP15SDISABLE_CPU2 | input | CP15SDISABLE (active high) |

6 FIFO Ports

No FIFO Ports in this model.

7 Parameters

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

| Name | Type | Description |
|------------------------|-------------|--|
| verbose | Boolean | Specify verbosity of output |
| showHiddenRegs | Boolean | Show hidden registers during register tracing |
| UAL | Boolean | Disassemble using UAL syntax |
| disableGICModel | Boolean | Disable the internal GIC model entirely |
| enableVFPAAtReset | Boolean | Enable vector floating point (SIMD and VFP) instructions at reset. (Enables cp10/11 in CPACR and sets FPEXC.EN) |
| enableSystemBus | Boolean | Add 32-bit artifact System bus port, allowing system registers to be externally implemented |
| enableSystemMonitorBus | Boolean | Add 32-bit artifact SystemMonitor bus port, allowing system register accesses to be externally monitored |
| compatibility | Enumeration | Specify compatibility mode ISA=0 gdb=1 nopSVC=2 |
| unpredictable | Enumeration | Specify unpredictable instruction behavior (undefined, nop, raz_wi, execute or assert) undefined=0 nop=1 raz_wi=2 execute=3 assert=4 |
| override_debugMask | Uns32 | Specifies debug mask, enabling debug output for model components |
| override_numCPUs | Uns32 | Specify the number of cores in a multiprocessor (maximum of 8 for GICv1/ GICv2) |
| override_affinityMask | Uns32 | Specify bitmask of implemented affinity bits in format Aff3:Aff2:Aff1:Aff0 (each a byte) |
| override_MPIDR_MT | Boolean | Specifies that processor is multithreaded |
| override_MPIDR_Aff0 | Uns32 | Override Aff0 field in MPIDR/MPIDR_EL1 register |

| | | |
|--------------------------------|---------|--|
| override_MPIDR_Aff1 | Uns32 | Override Aff1 field in MPIDR/MPIDR_EL1 register (also possible by writing CLUSTERIDAFF1 configuration net) |
| override_MPIDR_Aff2 | Uns32 | Override Aff2 field in MPIDR/MPIDR_EL1 register (also possible by writing CLUSTERIDAFF2 configuration net) |
| override_fcsePresent | Boolean | Specifies that FCSE is present (if true) |
| override_fpxcDexPresent | Boolean | Specifies that the FPEXC.DEX register field is implemented (if true) |
| override_advSIMDPresent | Boolean | Specifies that Advanced SIMD extensions are present (if true) |
| override_vfpPresent | Boolean | Specifies that VFP extensions are present (if true) |
| override_physicalBits | Uns32 | Specifies the implemented physical bus bits (defaults to connected physical bus width) |
| override_timerScaleFactor | Uns32 | Specifies the fraction of MIPS rate to use for MPCore timers (generic timers or global/local/watchdogs depending on implementation). Defaults to 20 for generic timers, 2 for others |
| override_GICD_NSACRPresent | Boolean | Specifies that optional GICD_NSACR distributor registers are present (GICv2 only) |
| override_GICD_PPISRPresent | Boolean | Specifies that implementation-specific GICD_PPISR distributor register is present (GICv1 ICDPPIS/ICPPISR, GICv1 and GICv2 only) |
| override_GICD_SPISRPresent | Boolean | Specifies that implementation-specific GICD_SPISR distributor registers are present (GICv1 ICDSPIS/ICSPISR) |
| override_GIC_PPIMask | Uns32 | Specify bitmask of implemented PPIs in the GIC (e.g. ID16 is 0x0001, ID31 is 0x8000) |
| override_GICCDISABLE | Boolean | Specify initial value of GICCDISABLE |
| override_SCTLR_V | Boolean | Override SCTLR.V with the passed value (enables high vectors) |
| override_SCTLR_CP15BEN_Present | Boolean | Enable ARMv7 SCTLR.CP15BEN bit (CP15 barrier enable) |
| override_MIDR | Uns32 | Override MIDR/MIDR_EL1 register |
| override_CTR | Uns32 | Override CTR/CTR_EL0 register |
| override_TLBTR | Uns32 | Override TLBTR register |
| override_CLIDR | Uns32 | Override CLIDR/CLIDR_EL1 register |
| override_AIDR | Uns32 | Override AIDR/AIDR_EL1 register |
| override_CBAR | Uns32 | Override Configuration Base Address Register (Corresponds to value on PERIPHBASE input pins) |
| override_PFR0 | Uns32 | Override ID_PFR0/ID_PFR0_EL1 register |
| override_PFR1 | Uns32 | Override ID_PFR1/ID_PFR1_EL1 register |
| override_DFR0 | Uns32 | Override ID_DFR0/ID_DFR0_EL1 register |

| | | |
|-----------------------------|-------|---|
| override_AFR0 | Uns32 | Override ID_AFR0/ID_AFR0_EL1 register |
| override_MMFR0 | Uns32 | Override ID_MMFR0/ID_MMFR0_EL1 register |
| override_MMFR1 | Uns32 | Override ID_MMFR1/ID_MMFR1_EL1 register |
| override_MMFR2 | Uns32 | Override ID_MMFR2/ID_MMFR2_EL1 register |
| override_MMFR3 | Uns32 | Override ID_MMFR3/ID_MMFR3_EL1 register |
| override_ISAR0 | Uns32 | Override ID_ISAR0/ID_ISAR0_EL1 register |
| override_ISAR1 | Uns32 | Override ID_ISAR1/ID_ISAR1_EL1 register |
| override_ISAR2 | Uns32 | Override ID_ISAR2/ID_ISAR2_EL1 register |
| override_ISAR3 | Uns32 | Override ID_ISAR3/ID_ISAR3_EL1 register |
| override_ISAR4 | Uns32 | Override ID_ISAR4/ID_ISAR4_EL1 register |
| override_ISAR5 | Uns32 | Override ID_ISAR5/ID_ISAR5_EL1 register |
| override_PMCR | Uns32 | Override PMCR/PMCR_EL0 register (not functionally significant in the model) |
| override_PMCEID0 | Uns64 | Override PMCEID0/PMCEID0_EL0 register (not functionally significant in the model) |
| override_PMCEID1 | Uns64 | Override PMCEID1/PMCEID1_EL0 register (not functionally significant in the model) |
| override_DBGDIDR | Uns32 | Override DBGDIDR register (not functionally significant in the model) |
| override_FPSID | Uns32 | Override SIMD/VFP FPSID register |
| override_MVFR0 | Uns32 | Override SIMD/VFP MVFR0/MVFR0_EL1 register |
| override_MVFR1 | Uns32 | Override SIMD/VFP MVFR1/MVFR1_EL1 register |
| override_FPEXC | Uns32 | Override SIMD/VFP FPEXC/FPEXC32_EL2 register |
| override_GICC_IIDR | Uns32 | Override GICC_IIDR register (GICv1 ICCIIDR) |
| override_GICD_TYPER | Uns32 | Override GICD_TYPER register (GICv1 ICDICTR) |
| override_GICD_TYPER_ITLines | Uns32 | Override ITLinesNumber field of GICD_TYPER register (GICv1 ICDICTR) |
| override_GICD_ICFGRN | Uns32 | Override reset value of GICD_ICFGR2...GICD_ICFGRn (GICv1 ICDICFR2...ICDICFRn) |
| override_GICD_IIDR | Uns32 | Override GICD_IIDR register (GICv1 ICDIIDR) |
| override_GICH_VTR | Uns32 | Override GICH_VTR register |
| override_ICCPMRBits | Uns32 | Specify the number of writable bits in GICC_PMR (GICv1 ICCPMR) |
| override_minICCBPR | Uns32 | Specify the minimum possible value for GICC_BPR (GICv1 ICCBPR) |

| | | |
|------------------------------|---------|--|
| override_ERG | Uns32 | Specifies exclusive reservation granule |
| override_CCSIDR_1I | Uns32 | Override CCSIDR/CCSIDR_EL1 (level 1 instruction) |
| override_CCSIDR_1D | Uns32 | Override CCSIDR/CCSIDR_EL1 (level 1 data) |
| override_CCSIDR_2I | Uns32 | Override CCSIDR/CCSIDR_EL1 (level 2 instruction) |
| override_CCSIDR_2D | Uns32 | Override CCSIDR/CCSIDR_EL1 (level 2 data) |
| override_CCSIDR_3I | Uns32 | Override CCSIDR/CCSIDR_EL1 (level 3 instruction) |
| override_CCSIDR_3D | Uns32 | Override CCSIDR/CCSIDR_EL1 (level 3 data) |
| override_CCSIDR_4I | Uns32 | Override CCSIDR/CCSIDR_EL1 (level 4 instruction) |
| override_CCSIDR_4D | Uns32 | Override CCSIDR/CCSIDR_EL1 (level 4 data) |
| override_CCSIDR_5I | Uns32 | Override CCSIDR/CCSIDR_EL1 (level 5 instruction) |
| override_CCSIDR_5D | Uns32 | Override CCSIDR/CCSIDR_EL1 (level 5 data) |
| override_CCSIDR_6I | Uns32 | Override CCSIDR/CCSIDR_EL1 (level 6 instruction) |
| override_CCSIDR_6D | Uns32 | Override CCSIDR/CCSIDR_EL1 (level 6 data) |
| override_CCSIDR_7I | Uns32 | Override CCSIDR/CCSIDR_EL1 (level 7 instruction) |
| override_CCSIDR_7D | Uns32 | Override CCSIDR/CCSIDR_EL1 (level 7 data) |
| override_STRoffsetPC12 | Boolean | Specifies that STR/STR of PC should do so with 12:byte offset from the current instruction (if true), otherwise an 8:byte offset is used |
| override_fcseRequiresMMU | Boolean | Specifies that FCSE is active only when MMU is enabled (if true) |
| override_ignoreBadCp15 | Boolean | Specifies whether invalid coprocessor 15 access should be ignored (if true) or cause Invalid Instruction exceptions (if false) |
| override_SGIDisable | Boolean | Override whether GIC SGIs may be disabled (if true) or are permanently enabled (if false) |
| override_condUndefined | Boolean | Force undefined instructions to take Undefined Instruction exception even if they are conditional |
| override_deviceStrongAligned | Boolean | Force accesses to Device and Strongly Ordered regions to be aligned |
| override_Control_V | Boolean | Override SCTL.R.V with the passed value (deprecated, use override_SCTL.R.V) |

| | | |
|---------------------------------|-------|---|
| override_MainId | Uns32 | Override MIDR register (deprecated, use override_MIDR) |
| override_CacheType | Uns32 | Override CTR register (deprecated, use override_CTR) |
| override_TLBType | Uns32 | Override TLBTR register (deprecated, use override_TLBTR) |
| override_InstructionAttributes0 | Uns32 | Override ID_ISAR0 register (deprecated, use override_ISAR0) |
| override_InstructionAttributes1 | Uns32 | Override ID_ISAR1 register (deprecated, use override_ISAR1) |
| override_InstructionAttributes2 | Uns32 | Override ID_ISAR2 register (deprecated, use override_ISAR2) |
| override_InstructionAttributes3 | Uns32 | Override ID_ISAR3 register (deprecated, use override_ISAR3) |
| override_InstructionAttributes4 | Uns32 | Override ID_ISAR4 register (deprecated, use override_ISAR4) |
| override_InstructionAttributes5 | Uns32 | Override ID_ISAR5 register (deprecated, use override_ISAR5) |

8 Execution Modes

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

| Name | Code |
|------------|------|
| User | 16 |
| FIQ | 17 |
| IRQ | 18 |
| Supervisor | 19 |
| Monitor | 22 |
| Abort | 23 |
| Undefined | 27 |
| System | 31 |

9 Exceptions

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

| Name | Code |
|-------------------|------|
| Reset | 0 |
| Undefined | 1 |
| SupervisorCall | 2 |
| SecureMonitorCall | 3 |
| PrefetchAbort | 5 |
| DataAbort | 6 |
| IRQ | 8 |

| | |
|-----|---|
| FIQ | 9 |
|-----|---|

10 Hierarchy of the model

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

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

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

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

10.1 Level 1: MPCORE

This level in the model hierarchy has 2 commands.

This level in the model hierarchy has no register groups.

This level in the model hierarchy has 3 children:

CPU0, CPU1 and CPU2

10.2 Level 2: CPU

This level in the model hierarchy has 5 commands.

This level in the model hierarchy has 20 register groups:

Table 7. Register groups

| Group name | Registers |
|-------------------------------|-----------|
| Core | 16 |
| Control | 3 |
| User | 7 |
| FIQ | 8 |
| IRQ | 3 |
| Supervisor | 3 |
| Monitor | 3 |
| Undefined | 3 |
| Abort | 3 |
| SIMD_VFP | 32 |
| SIMD_VFP_SYS | 5 |
| Coprocessor_32_bit | 122 |
| Coprocessor_32_bit_secure | 20 |
| Coprocessor_32_bit_non_secure | 20 |
| Integration_support | 24 |
| MPCore_SCR | 8 |
| MPCore_distributor | 91 |

| | |
|----------------------------|----|
| MPCore_processor_interface | 9 |
| MPCore_timer_and_watchdog | 10 |
| MPCore_global_timer | 5 |

This level in the model hierarchy has no children.

11 Model Commands

11.1 Level 1: MPCORE

11.1.1 *isync*

specify instruction address range for synchronous execution

Table 8. *isync* command arguments

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

11.1.2 *itrace*

enable or disable instruction tracing

Table 9. *itrace* command arguments

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

11.2 Level 2: CPU

11.2.1 *debugflags*

show or modify the processor debug flags

Table 10. *debugflags* command arguments

| Argument | Type | Description |
|----------|---------|---|
| -get | Boolean | print current processor flags value |
| -set | Int32 | new processor flags (only flags 0x000003e4 can be modified) |

11.2.2 *dumpTLB*

report TLB contents

Table 11. *dumpTLB* command arguments

| Argument | Type | Description |
|----------|---------|---|
| -all | Boolean | show the contents of all TLBs (if False, show just the current TLB) |

11.2.3 *isync*

specify instruction address range for synchronous execution

Table 12. *isync* command arguments

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

| | | |
|------------|-------|--|
| -addresslo | Uns64 | start address of synchronous execution range |
|------------|-------|--|

11.2.4 itrace

enable or disable instruction tracing

Table 13. itrace command arguments

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

11.2.5 validateTLB

check TLB contents against page tables in memory and report incoherent entries

Table 14. validateTLB command arguments

| Argument | Type | Description |
|----------|---------|---|
| -all | Boolean | check all TLBs (if False, validate just the current TLB) |
| -verbose | Boolean | show all TLB entries (if False, show only incoherent entries) |

12 Registers

12.1 Level 1: MPCORE

No registers.

12.2 Level 2: CPU

12.2.1 Core

Table 15. Registers at level 2, type: CPU, register group: 'Core'

| Name | Bits | Initial value (Hex) | | Description |
|------|------|---------------------|----|---------------|
| r0 | 32 | 0 | rw | |
| r1 | 32 | 0 | rw | |
| r2 | 32 | 0 | rw | |
| r3 | 32 | 0 | rw | |
| r4 | 32 | 0 | rw | |
| r5 | 32 | 0 | rw | |
| r6 | 32 | 0 | rw | |
| r7 | 32 | 0 | rw | |
| r8 | 32 | 0 | rw | |
| r9 | 32 | 0 | rw | |
| r10 | 32 | 0 | rw | |
| r11 | 32 | 0 | rw | frame pointer |

| | | | | |
|-----|----|---|----|-----------------|
| r12 | 32 | 0 | rw | |
| sp | 32 | 0 | rw | stack pointer |
| lr | 32 | 0 | rw | |
| pc | 32 | 0 | rw | program counter |

12.2.2 Control

Table 16. Registers at level 2, type: CPU, register group: 'Control'

| Name | Bits | Initial value (Hex) | | Description |
|------|------|---------------------|----|------------------------------|
| fps | 32 | 0 | rw | archaic FPSCR view (for gdb) |
| cpsr | 32 | 1d3 | rw | |
| spsr | 32 | 0 | rw | |

12.2.3 User

Table 17. Registers at level 2, type: CPU, register group: 'User'

| Name | Bits | Initial value (Hex) | | Description |
|---------|------|---------------------|----|-------------|
| r8_usr | 32 | 0 | rw | |
| r9_usr | 32 | 0 | rw | |
| r10_usr | 32 | 0 | rw | |
| r11_usr | 32 | 0 | rw | |
| r12_usr | 32 | 0 | rw | |
| sp_usr | 32 | 0 | rw | |
| lr_usr | 32 | 0 | rw | |

12.2.4 FIQ

Table 18. Registers at level 2, type: CPU, register group: 'FIQ'

| Name | Bits | Initial value (Hex) | | Description |
|----------|------|---------------------|----|-------------|
| r8_fiq | 32 | 0 | rw | |
| r9_fiq | 32 | 0 | rw | |
| r10_fiq | 32 | 0 | rw | |
| r11_fiq | 32 | 0 | rw | |
| r12_fiq | 32 | 0 | rw | |
| sp_fiq | 32 | 0 | rw | |
| lr_fiq | 32 | 0 | rw | |
| spsr_fiq | 32 | 0 | rw | |

12.2.5 IRQ

Table 19. Registers at level 2, type: CPU, register group: 'IRQ'

| Name | Bits | Initial value (Hex) | | Description |
|------|------|---------------------|--|-------------|
|------|------|---------------------|--|-------------|

| | | | | |
|----------|----|---|----|--|
| sp_irq | 32 | 0 | rw | |
| lr_irq | 32 | 0 | rw | |
| spsr_irq | 32 | 0 | rw | |

12.2.6 Supervisor

Table 20. Registers at level 2, type: CPU, register group: 'Supervisor'

| Name | Bits | Initial value (Hex) | | Description |
|----------|------|---------------------|----|-------------|
| sp_svc | 32 | 0 | rw | |
| lr_svc | 32 | 0 | rw | |
| spsr_svc | 32 | 0 | rw | |

12.2.7 Monitor

Table 21. Registers at level 2, type: CPU, register group: 'Monitor'

| Name | Bits | Initial value (Hex) | | Description |
|----------|------|---------------------|----|-------------|
| sp_mon | 32 | 0 | rw | |
| lr_mon | 32 | 0 | rw | |
| spsr_mon | 32 | 0 | rw | |

12.2.8 Undefined

Table 22. Registers at level 2, type: CPU, register group: 'Undefined'

| Name | Bits | Initial value (Hex) | | Description |
|------------|------|---------------------|----|-------------|
| sp_undef | 32 | 0 | rw | |
| lr_undef | 32 | 0 | rw | |
| spsr_undef | 32 | 0 | rw | |

12.2.9 Abort

Table 23. Registers at level 2, type: CPU, register group: 'Abort'

| Name | Bits | Initial value (Hex) | | Description |
|----------|------|---------------------|----|-------------|
| sp_abt | 32 | 0 | rw | |
| lr_abt | 32 | 0 | rw | |
| spsr_abt | 32 | 0 | rw | |

12.2.10 SIMD_VFP

Table 24. Registers at level 2, type: CPU, register group: 'SIMD_VFP'

| Name | Bits | Initial value (Hex) | | Description |
|------|------|---------------------|----|-------------|
| d0 | 64 | 0 | rw | |

| | | | | |
|-----|----|---|----|--|
| d1 | 64 | 0 | rw | |
| d2 | 64 | 0 | rw | |
| d3 | 64 | 0 | rw | |
| d4 | 64 | 0 | rw | |
| d5 | 64 | 0 | rw | |
| d6 | 64 | 0 | rw | |
| d7 | 64 | 0 | rw | |
| d8 | 64 | 0 | rw | |
| d9 | 64 | 0 | rw | |
| d10 | 64 | 0 | rw | |
| d11 | 64 | 0 | rw | |
| d12 | 64 | 0 | rw | |
| d13 | 64 | 0 | rw | |
| d14 | 64 | 0 | rw | |
| d15 | 64 | 0 | rw | |
| d16 | 64 | 0 | rw | |
| d17 | 64 | 0 | rw | |
| d18 | 64 | 0 | rw | |
| d19 | 64 | 0 | rw | |
| d20 | 64 | 0 | rw | |
| d21 | 64 | 0 | rw | |
| d22 | 64 | 0 | rw | |
| d23 | 64 | 0 | rw | |
| d24 | 64 | 0 | rw | |
| d25 | 64 | 0 | rw | |
| d26 | 64 | 0 | rw | |
| d27 | 64 | 0 | rw | |
| d28 | 64 | 0 | rw | |
| d29 | 64 | 0 | rw | |
| d30 | 64 | 0 | rw | |
| d31 | 64 | 0 | rw | |

12.2.11 SIMD_VFP_SYS

Table 25. Registers at level 2, type: CPU, register group: 'SIMD_VFP_SYS'

| Name | Bits | Initial value (Hex) | | Description |
|-------|------|---------------------|----|-------------------------------|
| FPSID | 32 | 41033092 | r- | floating-point system ID |
| FPSCR | 32 | 0 | rw | floating-point status/control |
| FPEXC | 32 | 0 | rw | floating-point exception |
| MVFR0 | 32 | 10110222 | r- | Media/VFP feature 0 |
| MVFR1 | 32 | 11111111 | r- | Media/VFP feature 1 |

12.2.12 Coprocessor_32_bit

Table 26. Registers at level 2, type: CPU, register group: 'Coprocessor_32_bit'

| Name | Bits | Initial value (Hex) | | Description |
|------------|------|---------------------|----|---|
| ACTLR | 32 | 0 | rw | Auxiliary Control |
| ADFSR | 32 | 0 | rw | Auxiliary Data Fault Status |
| AIDR | 32 | 0 | r- | Auxiliary ID |
| AIFSR | 32 | 0 | rw | Auxiliary Instruction Fault Status |
| ATS1CPR | 32 | - | -w | Address Translate Stage 1 Current State EL1 Read |
| ATS1CPW | 32 | - | -w | Address Translate Stage 1 Current State EL1 Write |
| ATS1CUR | 32 | - | -w | Address Translate Stage 1 Current State Unprivileged Read |
| ATS1CUW | 32 | - | -w | Address Translate Stage 1 Current State Unprivileged Write |
| ATS12NSOPR | 32 | - | -w | Address Translate Stages 1 and 2 Non-Secure Only EL1 Read |
| ATS12NSOPW | 32 | - | -w | Address Translate Stages 1 and 2 Non-Secure Only EL1 Write |
| ATS12NSOUR | 32 | - | -w | Address Translate Stages 1 and 2 Non-Secure Only Unprivileged Read |
| ATS12NSOUW | 32 | - | -w | Address Translate Stages 1 and 2 Non-Secure Only Unprivileged Write |
| BPIALL | 32 | - | -w | Branch Predictor Invalidate All |
| BPIALLIS | 32 | - | -w | Branch Predictor Invalidate All (IS) |
| BPIMVA | 32 | - | -w | Branch Predictor Invalidate by VA |
| CBAR | 32 | 13080000 | rw | Configuration Base Address |
| CCSIDR | 32 | 201fe019 | r- | Cache Size ID |
| CLIDR | 32 | 9200003 | r- | Cache Level ID |
| CONTEXTIDR | 32 | 0 | rw | Context ID |
| CP15DMB | 32 | - | -w | CP15 Data Memory Barrier |
| CP15DSB | 32 | - | -w | CP15 Data Synchronization Barrier |
| CP15ISB | 32 | - | -w | CP15 Instruction Synchronization Barrier |
| CP15NOP | 32 | - | -w | CP15 NOP |
| CPACR | 32 | 0 | rw | Coprocessor Access Control |
| CSSELR | 32 | 1 | rw | Cache Size Selection |
| CTR | 32 | 83338003 | r- | Cache Type |
| DACR | 32 | 0 | rw | Domain Access Control |
| DBGDIDR | 32 | 0 | r- | Debug ID |
| DCCIMVAC | 32 | - | -w | Data Cache Line Clean and Invalidate by VA to PoC |
| DCCISW | 32 | - | -w | Data Cache Line Clean and Invalidate by Set/Way |
| DCCMVAC | 32 | - | -w | Data Cache Line Clean by VA to PoC |
| DCCMVAU | 32 | - | -w | Data Cache Line Clean by VA to PoU |
| DCCSW | 32 | - | -w | Data Cache Line Clean by Set/Way |

| | | | | |
|-----------|----|----------|----|--|
| DCIMVAC | 32 | - | -w | Data Cache Line Invalidate by VA to PoC |
| DCISW | 32 | - | -w | Data Cache Line Invalidate by Set/Way |
| DFAR | 32 | 0 | rw | Data Fault Address |
| DFSR | 32 | 0 | rw | Data Fault Status |
| DTLBIALL | 32 | - | -w | Invalidate Entire Data TLB |
| DTLBIASID | 32 | - | -w | Invalidate Data TLB by ASID |
| DTLBIMVA | 32 | - | -w | Invalidate Data TLB by VA |
| DTLBIMVAA | 32 | - | -w | Invalidate Data TLB by VA, all ASID |
| DTLBLR | 32 | 0 | rw | TLB Lockdown |
| ICIALLU | 32 | - | -w | Instruction Cache Invalidate All |
| ICIALLUIS | 32 | - | -w | Instruction Cache Invalidate All (IS) |
| ICIMVAU | 32 | - | -w | Instruction Cache Invalidate by VA |
| ID_AFR0 | 32 | 0 | r- | Auxiliary Feature 0 |
| ID_DFR0 | 32 | 0 | r- | Debug Feature 0 |
| ID_ISAR0 | 32 | 101111 | r- | Instruction Set Attribute 0 |
| ID_ISAR1 | 32 | 13112111 | r- | Instruction Set Attribute 1 |
| ID_ISAR2 | 32 | 21232041 | r- | Instruction Set Attribute 2 |
| ID_ISAR3 | 32 | 11112131 | r- | Instruction Set Attribute 3 |
| ID_ISAR4 | 32 | 11142 | r- | Instruction Set Attribute 4 |
| ID_ISAR5 | 32 | 0 | r- | Instruction Set Attribute 5 |
| ID_MMFR0 | 32 | 100103 | r- | Memory Model Feature 0 |
| ID_MMFR1 | 32 | 20000000 | r- | Memory Model Feature 1 |
| ID_MMFR2 | 32 | 1230000 | r- | Memory Model Feature 2 |
| ID_MMFR3 | 32 | 102111 | r- | Memory Model Feature 3 |
| ID_PFR0 | 32 | 1231 | r- | Processor Feature 0 |
| ID_PFR1 | 32 | 11 | r- | Processor Feature 1 |
| IFAR | 32 | 0 | rw | Instruction Fault Address |
| IFSR | 32 | 0 | rw | Instruction Fault Status |
| ISR | 32 | 0 | r- | Interrupt Status |
| ITLBIALL | 32 | - | -w | Invalidate Entire Instruction TLB |
| ITLBIASID | 32 | - | -w | Invalidate Instruction TLB by ASID |
| ITLBIMVA | 32 | - | -w | Invalidate Instruction TLB by VA |
| ITLBIMVAA | 32 | - | -w | Invalidate Instruction TLB by VA, all ASID |
| JIDR | 32 | 0 | rw | Jazelle ID |
| JMCR | 32 | 0 | rw | Jazelle Main Configuration |
| JOSCR | 32 | 0 | rw | Jazelle OS Control |
| MIDR | 32 | 411fc090 | r- | Main ID |
| MPIDR | 32 | 80000000 | r- | Multiprocessor Affinity |
| MVBAR | 32 | 0 | rw | Monitor Vector Base Address |
| NEONB | 32 | 0 | r- | NEON Busy |
| NMRR | 32 | 44e048e0 | rw | Normal Memory Remap |
| NSACR | 32 | 0 | rw | Non-Secure Access Control |

| | | | | |
|------------|----|----------|----|--|
| PAR | 32 | 0 | rw | Physical Address |
| PCR | 32 | 200 | rw | Power Control |
| PLEASR | 32 | 0 | r- | PLE Activity Status |
| PLEFSR | 32 | 0 | r- | PLE FIFO Status |
| PLEIDR | 32 | 0 | r- | PLE ID |
| PMCCNTR | 32 | 0 | rw | Performance Monitors Cycle Count |
| PMCNTENCLR | 32 | 0 | rw | Performance Monitors Count Enable Clear |
| PMCNTENSET | 32 | 0 | rw | Performance Monitors Count Enable Set |
| PMCR | 32 | 41093000 | rw | Performance Monitors Control |
| PMINTENCLR | 32 | 0 | rw | Performance Monitors Interrupt Enable Clear |
| PMINTENSET | 32 | 0 | rw | Performance Monitors Interrupt Enable Set |
| PMOVSr | 32 | 0 | rw | Performance Monitors Overflow Flag Status |
| PMSELR | 32 | 0 | rw | Performance Monitors Event Counter Selection |
| PMSWINC | 32 | - | -w | Performance Monitors Software Increment |
| PMUSERENR | 32 | 0 | rw | Performance Monitors User Enable |
| PMXEVcntr | 32 | 0 | rw | Performance Monitors Selected Event Count |
| PMXEVtyper | 32 | 0 | rw | Performance Monitors Selected Event Type |
| PRRR | 32 | 98aa4 | rw | Primary Region Remap |
| SCR | 32 | 0 | rw | Secure Configuration |
| SCTLR | 32 | c50078 | rw | System Control |
| SDER | 32 | 0 | rw | Secure Debug Enable |
| TCMTR | 32 | 0 | r- | TCM Type |
| TEECR | 32 | 0 | rw | T32EE Configuration |
| TEEHBR | 32 | 0 | rw | T32EE Handler Base |
| TLBIALL | 32 | - | -w | Invalidate Entire Unified TLB |
| TLBIALLIS | 32 | - | -w | Invalidate Entire Unified TLB (IS) |
| TLBIASID | 32 | - | -w | Invalidate Unified TLB by ASID |
| TLBIASIDIS | 32 | - | -w | Invalidate Unified TLB by ASID (IS) |
| TLBIMVA | 32 | - | -w | Invalidate Unified TLB by VA |
| TLBIMVAA | 32 | - | -w | Invalidate Unified TLB by VA, all ASID |
| TLBIMVAAIS | 32 | - | -w | Invalidate Unified TLB by VA, all ASID (IS) |
| TLBIMVAIS | 32 | - | -w | Invalidate Unified TLB by VA (IS) |
| TLBLDATTR | 32 | 0 | rw | TLB Lockdown Attributes |
| TLBLDPA | 32 | 0 | rw | TLB Lockdown PA |
| TLBLDRI | 32 | - | -w | TLB Lockdown Read Index |
| TLBLDVA | 32 | 0 | rw | TLB Lockdown VA |
| TLBLDWI | 32 | - | -w | TLB Lockdown Write Index |
| TLBTR | 32 | 400 | r- | TLB Type |
| TPIDRPRW | 32 | 0 | rw | PL0 Read/Write Software Thread ID |
| TPIDRURO | 32 | 0 | rw | PL0 Read-Only Software Thread ID |
| TPIDRURW | 32 | 0 | rw | PL1 Software Thread ID |
| TTBCR | 32 | 0 | rw | Translation Table Base Control |

| | | | | |
|-------|----|---|----|--------------------------|
| TTBR0 | 32 | 0 | rw | Translation Table Base 0 |
| TTBR1 | 32 | 0 | rw | Translation Table Base 1 |
| VBAR | 32 | 0 | rw | Vector Base Address |
| VCR | 32 | 0 | rw | Virtualization Control |
| VIR | 32 | 0 | rw | Virtualization Interrupt |

12.2.13 Coprocessor_32_bit_secure

Table 27. Registers at level 2, type: CPU, register group: 'Coprocessor_32_bit_secure'

| Name | Bits | Initial value (Hex) | | Description |
|--------------|------|---------------------|----|------------------------------------|
| ADFSR_S | 32 | 0 | rw | Auxiliary Data Fault Status |
| AIFSR_S | 32 | 0 | rw | Auxiliary Instruction Fault Status |
| CONTEXTIDR_S | 32 | 0 | rw | Context ID |
| CSSELR_S | 32 | 1 | rw | Cache Size Selection |
| DACR_S | 32 | 0 | rw | Domain Access Control |
| DFAR_S | 32 | 0 | rw | Data Fault Address |
| DFSR_S | 32 | 0 | rw | Data Fault Status |
| IFAR_S | 32 | 0 | rw | Instruction Fault Address |
| IFSR_S | 32 | 0 | rw | Instruction Fault Status |
| NMRR_S | 32 | 44e048e0 | rw | Normal Memory Remap |
| PAR_S | 32 | 0 | rw | Physical Address |
| PRRR_S | 32 | 98aa4 | rw | Primary Region Remap |
| SCTLR_S | 32 | c50078 | rw | System Control |
| TPIDRPRW_S | 32 | 0 | rw | PL0 Read/Write Software Thread ID |
| TPIDRURO_S | 32 | 0 | rw | PL0 Read-Only Software Thread ID |
| TPIDRURW_S | 32 | 0 | rw | PL1 Software Thread ID |
| TTBCR_S | 32 | 0 | rw | Translation Table Base Control |
| TTBR0_S | 32 | 0 | rw | Translation Table Base 0 |
| TTBR1_S | 32 | 0 | rw | Translation Table Base 1 |
| VBAR_S | 32 | 0 | rw | Vector Base Address |

12.2.14 Coprocessor_32_bit_non_secure

Table 28. Registers at level 2, type: CPU, register group: 'Coprocessor_32_bit_non_secure'

| Name | Bits | Initial value (Hex) | | Description |
|---------------|------|---------------------|----|------------------------------------|
| ADFSR_NS | 32 | 0 | rw | Auxiliary Data Fault Status |
| AIFSR_NS | 32 | 0 | rw | Auxiliary Instruction Fault Status |
| CONTEXTIDR_NS | 32 | 0 | rw | Context ID |
| CSSELR_NS | 32 | 1 | rw | Cache Size Selection |
| DACR_NS | 32 | 0 | rw | Domain Access Control |
| DFAR_NS | 32 | 0 | rw | Data Fault Address |
| DFSR_NS | 32 | 0 | rw | Data Fault Status |

| | | | | |
|-------------|----|----------|----|-----------------------------------|
| IFAR_NS | 32 | 0 | rw | Instruction Fault Address |
| IFSR_NS | 32 | 0 | rw | Instruction Fault Status |
| NMRR_NS | 32 | 44e048e0 | rw | Normal Memory Remap |
| PAR_NS | 32 | 0 | rw | Physical Address |
| PRRR_NS | 32 | 98aa4 | rw | Primary Region Remap |
| SCTLR_NS | 32 | c50078 | rw | System Control |
| TPIDRPRW_NS | 32 | 0 | rw | PL0 Read/Write Software Thread ID |
| TPIDRURO_NS | 32 | 0 | rw | PL0 Read-Only Software Thread ID |
| TPIDRURW_NS | 32 | 0 | rw | PL1 Software Thread ID |
| TTBCR_NS | 32 | 0 | rw | Translation Table Base Control |
| TTBR0_NS | 32 | 0 | rw | Translation Table Base 0 |
| TTBR1_NS | 32 | 0 | rw | Translation Table Base 1 |
| VBAR_NS | 32 | 0 | rw | Vector Base Address |

12.2.15 Integration_support

Table 29. Registers at level 2, type: CPU, register group: 'Integration_support'

| Name | Bits | Initial value (Hex) | | Description |
|-----------------|------|---------------------|----|---|
| transactPL | 32 | 1 | r- | privilege level of current memory transaction |
| transactAT | 32 | 0 | r- | current memory transaction type: PA=1, VA=0 |
| artifactPAR | 64 | 0 | r- | result of address translation for artifact write to ATS1CPR etc |
| PTWBankValid | 8 | 0 | r- | bitmask of valid banks (0x01 is stage 1, 0x02 is stage 2, 0x04-0x20 are stage 2 walks initiated by stage 1 level 0-3 entry lookups, respectively) |
| PTWAddressValid | 8 | 0 | r- | bitmask of valid bits for each of PTWAddressL0...PTWAddressL3, PTWBase, PTWInput and PTWOutput in current bank |
| PTWValueValid | 8 | 0 | r- | bitmask of valid bits for each of PTWValueL0...PTWValueL3 in current bank |
| PTWAddressL0 | 64 | 0 | r- | current bank PTW address, level 0 |
| PTWAddressL1 | 64 | 0 | r- | current bank PTW address, level 1 |
| PTWAddressL2 | 64 | 0 | r- | current bank PTW address, level 2 |
| PTWAddressL3 | 64 | 0 | r- | current bank PTW address, level 3 |
| PTWValueL0 | 64 | 0 | r- | current bank PTW value, level 0 |
| PTWValueL1 | 64 | 0 | r- | current bank PTW value, level 1 |
| PTWValueL2 | 64 | 0 | r- | current bank PTW value, level 2 |
| PTWValueL3 | 64 | 0 | r- | current bank PTW value, level 3 |
| PTWBase | 64 | 0 | r- | current bank PTW table base address |
| PTWInput | 64 | 0 | r- | current bank PTW input address |
| PTWOutput | 64 | 0 | r- | current bank PTW output address |
| PTWI_EL1S | 64 | - | -w | perform EL1(S) stage 1 page table walk for fetch, filling PTW query registers |

| | | | | |
|--------------|----|---|----|---|
| PTWD_EL1S | 64 | - | -w | perform EL1(S) stage 1 page table walk for load/store, filling PTW query registers |
| PTWI_EL1NS | 64 | - | -w | perform EL1(NS) stage 1 page table walk for fetch, filling PTW query registers |
| PTWD_EL1NS | 64 | - | -w | perform EL1(NS) stage 1 page table walk for load/store, filling PTW query registers |
| PTWI_current | 64 | - | -w | perform current mode page table walk for fetch, filling PTW query registers |
| PTWD_current | 64 | - | -w | perform current mode page table walk for load/store, filling PTW query registers |
| HaltReason | 8 | 0 | r- | bit field indicating halt reason |

12.2.16 MPCore_SCR

Table 30. Registers at level 2, type: CPU, register group: 'MPCore_SCR'

| Name | Bits | Initial value (Hex) | | Description |
|---------------------|------|---------------------|----|-------------------------------|
| SCUCPUPowerStatus | 32 | 3000000 | rw | SCU CPU Power Status |
| SCUConfiguration | 32 | 1502 | r- | SCU Configuration |
| SCUControl | 32 | 0 | rw | SCU Control |
| SCUFilteringEnd | 32 | 0 | rw | SCU Filtering End |
| SCUFilteringStart | 32 | 0 | rw | SCU Filtering Start |
| SCUInvalidateSecure | 32 | 0 | -w | SCU Invalidate Secure |
| SCUNSAC | 32 | 0 | rw | SCU Non-secure Access Control |
| SCUSAC | 32 | f | rw | SCU Secure Access Control |

12.2.17 MPCore_distributor

Table 31. Registers at level 2, type: CPU, register group: 'MPCore_distributor'

| Name | Bits | Initial value (Hex) | | Description |
|----------|------|---------------------|----|---------------------------|
| ICCIDR0 | 32 | d | r- | Component ID 0 |
| ICCIDR1 | 32 | f0 | r- | Component ID 1 |
| ICCIDR2 | 32 | 5 | r- | Component ID 2 |
| ICCIDR3 | 32 | b1 | r- | Component ID 3 |
| ICDABR0 | 32 | 0 | rw | Interrupt Set-Active 0 |
| ICDABR1 | 32 | 0 | rw | Interrupt Set-Active 1 |
| ICDABR2 | 32 | 0 | rw | Interrupt Set-Active 2 |
| ICDDCR | 32 | 0 | rw | Distributor Control |
| ICDICER0 | 32 | ffff | rw | Interrupt Clear-Enable 0 |
| ICDICER1 | 32 | 0 | rw | Interrupt Clear-Enable 1 |
| ICDICER2 | 32 | 0 | rw | Interrupt Clear-Enable 2 |
| ICDICFR0 | 32 | aaaaaaaa | rw | Interrupt Configuration 0 |
| ICDICFR1 | 32 | 7dc00000 | rw | Interrupt Configuration 1 |
| ICDICFR2 | 32 | 55555555 | rw | Interrupt Configuration 2 |

| | | | | |
|----------|----|----------|----|-------------------------------|
| ICDICFR3 | 32 | 55555555 | rw | Interrupt Configuration 3 |
| ICDICFR4 | 32 | 55555555 | rw | Interrupt Configuration 4 |
| ICDICFR5 | 32 | 55555555 | rw | Interrupt Configuration 5 |
| ICDICPR0 | 32 | 0 | rw | Interrupt Clear-Pending 0 |
| ICDICPR1 | 32 | 0 | rw | Interrupt Clear-Pending 1 |
| ICDICPR2 | 32 | 0 | rw | Interrupt Clear-Pending 2 |
| ICDICTR | 32 | fc42 | r- | Interrupt Controller Type |
| ICDIIDR | 32 | 102043b | r- | Distributor Implementor ID |
| ICDIPR0 | 32 | 0 | rw | Interrupt Priority 0 |
| ICDIPR1 | 32 | 0 | rw | Interrupt Priority 1 |
| ICDIPR2 | 32 | 0 | rw | Interrupt Priority 2 |
| ICDIPR3 | 32 | 0 | rw | Interrupt Priority 3 |
| ICDIPR4 | 32 | 0 | rw | Interrupt Priority 4 |
| ICDIPR5 | 32 | 0 | rw | Interrupt Priority 5 |
| ICDIPR6 | 32 | 0 | rw | Interrupt Priority 6 |
| ICDIPR7 | 32 | 0 | rw | Interrupt Priority 7 |
| ICDIPR8 | 32 | 0 | rw | Interrupt Priority 8 |
| ICDIPR9 | 32 | 0 | rw | Interrupt Priority 9 |
| ICDIPR10 | 32 | 0 | rw | Interrupt Priority 10 |
| ICDIPR11 | 32 | 0 | rw | Interrupt Priority 11 |
| ICDIPR12 | 32 | 0 | rw | Interrupt Priority 12 |
| ICDIPR13 | 32 | 0 | rw | Interrupt Priority 13 |
| ICDIPR14 | 32 | 0 | rw | Interrupt Priority 14 |
| ICDIPR15 | 32 | 0 | rw | Interrupt Priority 15 |
| ICDIPR16 | 32 | 0 | rw | Interrupt Priority 16 |
| ICDIPR17 | 32 | 0 | rw | Interrupt Priority 17 |
| ICDIPR18 | 32 | 0 | rw | Interrupt Priority 18 |
| ICDIPR19 | 32 | 0 | rw | Interrupt Priority 19 |
| ICDIPR20 | 32 | 0 | rw | Interrupt Priority 20 |
| ICDIPR21 | 32 | 0 | rw | Interrupt Priority 21 |
| ICDIPR22 | 32 | 0 | rw | Interrupt Priority 22 |
| ICDIPR23 | 32 | 0 | rw | Interrupt Priority 23 |
| ICDIPTR0 | 32 | 1010101 | rw | Interrupt Processor Targets 0 |
| ICDIPTR1 | 32 | 1010101 | rw | Interrupt Processor Targets 1 |
| ICDIPTR2 | 32 | 1010101 | rw | Interrupt Processor Targets 2 |
| ICDIPTR3 | 32 | 1010101 | rw | Interrupt Processor Targets 3 |
| ICDIPTR4 | 32 | 0 | rw | Interrupt Processor Targets 4 |
| ICDIPTR5 | 32 | 0 | rw | Interrupt Processor Targets 5 |
| ICDIPTR6 | 32 | 1000000 | rw | Interrupt Processor Targets 6 |
| ICDIPTR7 | 32 | 1010101 | rw | Interrupt Processor Targets 7 |
| ICDIPTR8 | 32 | 0 | rw | Interrupt Processor Targets 8 |
| ICDIPTR9 | 32 | 0 | rw | Interrupt Processor Targets 9 |

| | | | | |
|-----------|----|------|----|--------------------------------|
| ICDIPTR10 | 32 | 0 | rw | Interrupt Processor Targets 10 |
| ICDIPTR11 | 32 | 0 | rw | Interrupt Processor Targets 11 |
| ICDIPTR12 | 32 | 0 | rw | Interrupt Processor Targets 12 |
| ICDIPTR13 | 32 | 0 | rw | Interrupt Processor Targets 13 |
| ICDIPTR14 | 32 | 0 | rw | Interrupt Processor Targets 14 |
| ICDIPTR15 | 32 | 0 | rw | Interrupt Processor Targets 15 |
| ICDIPTR16 | 32 | 0 | rw | Interrupt Processor Targets 16 |
| ICDIPTR17 | 32 | 0 | rw | Interrupt Processor Targets 17 |
| ICDIPTR18 | 32 | 0 | rw | Interrupt Processor Targets 18 |
| ICDIPTR19 | 32 | 0 | rw | Interrupt Processor Targets 19 |
| ICDIPTR20 | 32 | 0 | rw | Interrupt Processor Targets 20 |
| ICDIPTR21 | 32 | 0 | rw | Interrupt Processor Targets 21 |
| ICDIPTR22 | 32 | 0 | rw | Interrupt Processor Targets 22 |
| ICDIPTR23 | 32 | 0 | rw | Interrupt Processor Targets 23 |
| ICDISER0 | 32 | ffff | rw | Interrupt Set-Enable 0 |
| ICDISER1 | 32 | 0 | rw | Interrupt Set-Enable 1 |
| ICDISER2 | 32 | 0 | rw | Interrupt Set-Enable 2 |
| ICDISPR0 | 32 | 0 | rw | Interrupt Set-Pending 0 |
| ICDISPR1 | 32 | 0 | rw | Interrupt Set-Pending 1 |
| ICDISPR2 | 32 | 0 | rw | Interrupt Set-Pending 2 |
| ICDISR0 | 32 | 0 | rw | Interrupt Group 0 |
| ICDISR1 | 32 | 0 | rw | Interrupt Group 1 |
| ICDISR2 | 32 | 0 | rw | Interrupt Group 2 |
| ICDPPIS | 32 | 0 | r- | PPI STATUS |
| ICDSGIR | 32 | 0 | -w | Software-Generated Interrupt |
| ICDSPIS0 | 32 | 0 | r- | SPI Status 0 |
| ICDSPIS1 | 32 | 0 | r- | SPI Status 1 |
| ICPIDR0 | 32 | 90 | r- | Peripheral ID 0 |
| ICPIDR1 | 32 | b3 | r- | Peripheral ID 1 |
| ICPIDR2 | 32 | 1b | r- | Peripheral ID 2 |
| ICPIDR3 | 32 | 0 | r- | Peripheral ID 3 |
| ICPIDR4 | 32 | 4 | r- | Peripheral ID 4 |
| ICPIDR5 | 32 | 0 | r- | Peripheral ID 5 |
| ICPIDR6 | 32 | 0 | r- | Peripheral ID 6 |
| ICPIDR7 | 32 | 0 | r- | Peripheral ID 7 |

12.2.18 MPCore_processor_interface

Table 32. Registers at level 2, type: CPU, register group: 'MPCore_processor_interface'

| Name | Bits | Initial value (Hex) | | Description |
|---------|------|---------------------|----|----------------------|
| ICCABPR | 32 | 3 | rw | Aliased Binary Point |
| ICCBPR | 32 | 2 | rw | Binary Point |

| | | | | |
|---------|----|----------|----|------------------------------------|
| ICCEOIR | 32 | 0 | -w | End of Interrupt |
| ICCHPIR | 32 | 3ff | r- | Highest Priority Pending Interrupt |
| ICCIAR | 32 | 3ff | r- | Interrupt Acknowledge |
| ICCICR | 32 | 0 | rw | CPU Interface Control |
| ICCIIDR | 32 | 3901243b | r- | CPU Interface ID |
| ICCPMR | 32 | 0 | rw | Interrupt Priority Mask |
| ICCRPR | 32 | ff | r- | Running Priority |

12.2.19 MPCore_timer_and_watchdog

Table 33. Registers at level 2, type: CPU, register group: 'MPCore_timer_and_watchdog'

| Name | Bits | Initial value (Hex) | | Description |
|-------------------|------|---------------------|----|---------------------------------|
| PTControl | 32 | 0 | rw | Private Timer Control |
| PTCounter | 32 | 0 | rw | Private Timer Counter |
| PTInterruptStatus | 32 | 0 | rw | Private Timer Interrupt Status |
| PTLoad | 32 | 0 | rw | Private Timer Load |
| WTControl | 32 | 0 | rw | Watchdog Timer Control |
| WTCOUNTER | 32 | 0 | rw | Watchdog Timer Counter |
| WTDisable | 32 | 0 | -w | Watchdog Timer Disable |
| WTInterruptStatus | 32 | 0 | rw | Watchdog Timer Interrupt Status |
| WTLoad | 32 | 0 | rw | Watchdog Timer Load |
| WTRResetStatus | 32 | 0 | rw | Watchdog Timer Reset Status |

12.2.20 MPCore_global_timer

Table 34. Registers at level 2, type: CPU, register group: 'MPCore_global_timer'

| Name | Bits | Initial value (Hex) | | Description |
|-------------------|------|---------------------|----|-------------------------------|
| GTAutoIncrement | 32 | 0 | rw | Global Timer Auto-Increment |
| GTComparator | 64 | 0 | rw | Global Timer Comparator |
| GTControl | 32 | 0 | rw | Global Timer Control |
| GTCounter | 64 | 0 | rw | Global Timer Counter |
| GTInterruptStatus | 32 | 0 | rw | Global Timer Interrupt Status |

#