ARM® Cortex®-A53 MPCore Processor Advanced SIMD and Floating-point Extension

Revision: r0p3

Technical Reference Manual

ARM DDI 0502E (ID062314)

ARM Cortex-A53 MPCore Processor Advanced SIMD and Floating-point Extension
Technical Reference Manual

Release Information

The following changes have been made to this book.

Change history

Date	Issue	Confidentiality	Change
09 August 2013	A	Confidential	Release for r0p0
04 November 2013	B	Confidential	Release for r0p1
13 December 2013	C	Confidential	Release for r0p2
14 February 2014	D	Non-Confidential	Second release for r0p2
30 April 2014	E	Non-Confidential	Release for r0p3

Product Status

The information in this document is final, that is for a developed product.

Web Address

`http://www.arm.com`

Contents
ARM Cortex-A53 MPCore Processor Advanced SIMD and Floating-point Extension Technical Reference Manual

Preface

This preface introduces the *ARM® Cortex®-A53 MPCore Processor Advanced SIMD and Floating-point Extension Technical Reference Manual*. It contains the following sections:

- *About this book* on page v.
- *Feedback* on page vii.

About this book

This book is for the Cortex-A53 processor Advanced SIMD and Floating-point Extension.

Product revision status

The r*m*p*n* identifier indicates the revision status of the product described in this book, for example, r1p2, where:

r*m* Identifies the major revision of the product, for example, r1.

p*n* Identifies the minor revision or modification status of the product, for example, p2.

Intended audience

This book is written for system designers, system integrators, and programmers who are designing or programming a *System-on-Chip* (SoC) that uses the Cortex-A53 processor with the optional Advanced SIMD and Floating-point Extension.

Using this book

This book is organized into the following chapters:

Chapter 1 *Introduction*

Read this for an introduction to the optional Cortex-A53 Advanced SIMD and Floating-point Extension, and a description of its features.

Chapter 2 *Programmers Model*

Read this for a description of the programmers model for the Cortex-A53 Advanced SIMD and Floating-point Extension.

Appendix A *Revisions*

Read this for a description of the technical changes between released issues of this book.

Glossary

The *ARM® Glossary* is a list of terms used in ARM documentation, together with definitions for those terms. The *ARM® Glossary* does not contain terms that are industry standard unless the ARM meaning differs from the generally accepted meaning.

See ARM® Glossary `http://infocenter.arm.com/help/topic/com.arm.doc.aeg0014-/index.html`.

Conventions

This book uses the conventions that are described in:

- *Typographical conventions* on page vi.

Typographical conventions

The following table describes the typographical conventions:

Typographical conventions

Style	Purpose
italic	Introduces special terminology, denotes cross-references, and citations.
bold	Highlights interface elements, such as menu names. Denotes signal names. Also used for terms in descriptive lists, where appropriate.
`monospace`	Denotes text that you can enter at the keyboard, such as commands, file and program names, and source code.
`monospace`	Denotes a permitted abbreviation for a command or option. You can enter the underlined text instead of the full command or option name.
`monospace` *`italic`*	Denotes arguments to monospace text where the argument is to be replaced by a specific value.
`monospace bold`	Denotes language keywords when used outside example code.
<and>	Encloses replaceable terms for assembler syntax where they appear in code or code fragments. For example: `MRC p15, 0 <Rd>, <CRn>, <CRm>, <Opcode_2>`
SMALL CAPITALS	Used in body text for a few terms that have specific technical meanings, that are defined in the *ARM® glossary*. For example, IMPLEMENTATION DEFINED, IMPLEMENTATION SPECIFIC, UNKNOWN, and UNPREDICTABLE.

Additional reading

This section lists publications by ARM and by third parties.

See Infocenter http://infocenter.arm.com, for access to ARM documentation.

ARM publications

This book contains information that is specific to this product. See the following documents for other relevant information:

- *ARM® Cortex®-A53 MPCore Processor Technical Reference Manual* (ARM DDI 0500).
- *ARM® Architecture Reference Manual, ARMv8, for ARMv8-A architecture profile* (ARM DDI 0487).
- *ARM® Cortex®-A Series Programmer's Guide* (ARM DEN0013B).

The following confidential books are only available to licensees:

- *ARM® Cortex®-A53 MPCore Processor Cryptography Extension Technical Reference Manual* (ARM DDI 0501).
- *ARM® Cortex®-A53 MPCore Processor Configuration and Sign-off Guide* (ARM DII 0281).
- *ARM® Cortex®-A53 MPCore Processor Integration Manual* (ARM DIT 0036).

Other publications

This section lists relevant documents published by third parties:

- *ANSI/IEEE Std 754-2008, IEEE Standard for Floating-Point Arithmetic*.

Feedback

ARM welcomes feedback on this product and its documentation.

Feedback on this product

If you have any comments or suggestions about this product, contact your supplier and give:

- The product name.
- The product revision or version.
- An explanation with as much information as you can provide. Include symptoms and diagnostic procedures if appropriate.

Feedback on content

If you have comments on content then send an e-mail to `errata@arm.com`. Give:

- The title.
- The number, ARM DDI 0502E.
- The page numbers to which your comments apply.
- A concise explanation of your comments.

ARM also welcomes general suggestions for additions and improvements.

Note

ARM tests the PDF only in Adobe Acrobat and Acrobat Reader, and cannot guarantee the quality of the represented document when used with any other PDF reader.

Chapter 1
Introduction

This chapter introduces the optional Advanced SIMD and Floating-point Extension. It contains the following sections:

- *About the Cortex-A53 Advanced SIMD and Floating-point Extension* on page 1-2.
- *Floating-point support* on page 1-3.

1.1 About the Cortex-A53 Advanced SIMD and Floating-point Extension

The Cortex-A53 processor supports the Advanced SIMD and Scalar Floating-point instructions in the A64 instruction set, and the Advanced SIMD and VFP instructions in the A32 and T32 instruction sets.

The ARMv8 architecture eliminates the concept of version numbers for Advanced SIMD and Floating-point in the AArch64 execution state.

1.2 Floating-point support

The Cortex-A53 floating-point implementation:

- Does not generate floating-point exceptions.
- Implements all scalar operations in hardware, with support for all combinations of:
  - Rounding modes.
  - Flush-to-zero.
  - Default *Not a Number* (NaN) modes.

Chapter 2
Programmers Model

This section describes the programmers model for the Cortex-A53 processor Advanced SIMD and Floating-point Extension in:

2.1 Accessing the feature identification registers

Software can identify the Advanced SIMD and Floating-point features using the feature identification registers in the AArch64 and AArch32 execution states.

You can access the feature identification registers in the AArch64 execution state using the `MRS` instructions, for example:

```
MRS <Xt>, ID_AA64PFR0_EL1 ; Read ID_AA64PFR0_EL1 into Xt
MRS <Xt>, MVFR0_EL1       ; Read MVFR0_EL1 into Xt
MRS <Xt>, MVFR1_EL1       ; Read MVFR1_EL1 into Xt
MRS <Xt>, MVFR2_EL1       ; Read MVFR2_EL1 into Xt
```

You can access the feature identification registers in the AArch32 execution state using the `VMRS` instructions, for example:

```
VMRS <Rt>, FPSID ; Read FPSID into Rt
VMRS <Rt>, MVFR0 ; Read MVFR0 into Rt
VMRS <Rt>, MVFR1 ; Read MFFR1 into Rt
VMRS <Rt>, MVFR2 ; Read MVFR2 into Rt
```

Table 2-1 lists the feature identification registers for the Advanced SIMD and Floating-point Extension.

Table 2-1 Advanced SIMD and Scalar Floating-point feature identification registers

AArch64 name	AArch32 name	Description
ID_AA64PFR0_EL1	-	Gives additional information about implemented processor features in AArch64. See the *ARM® Cortex®-A53 MPCore Processor Technical Reference Manual*.
-	FPSID	See *Floating-Point System ID Register* on page 2-14.
MVFR0_EL1	MVFR0	See: • *Media and Floating-point Feature Register 0* on page 2-7 (AArch64). • *Media and Floating-point Feature Register 0* on page 2-17 (AArch32).
MVFR1_EL1	MVFR1	See: • *Media and Floating-point Feature Register 1* on page 2-8 (AArch64). • *Media and Floating-point Feature Register 1* on page 2-18 (AArch32).
MVFR2_EL1	MVFR2	See: • *Media and Floating-point Feature Register 2* on page 2-9 (AArch64). • *Media and Floating-point Feature Register 2* on page 2-20 (AArch32).

2.2 AArch64 register summary

Table 2-2 gives a summary of the Cortex-A53 processor Advanced SIMD and Floating-point system registers in the AArch64 execution state.

Table 2-2 AArch64 Advanced SIMD and Floating-point system registers

Name	Type	Reset	Description
FPCR	RW	`0x00000000`	See *Floating-point Control Register* on page 2-4
FPSR	RW	`0x00000000`	See *Floating-point Status Register* on page 2-5
MVFR0_EL1	RO	`0x10110222`	See *Media and Floating-point Feature Register 0* on page 2-7
MVFR1_EL1	RO	`0x12111111`	See *Media and Floating-point Feature Register 1* on page 2-8
MVFR2_EL1	RW	`0x00000043`	See *Media and Floating-point Feature Register 2* on page 2-9
FPEXC32_EL2	RW	`0x00000700`	See *Floating-point Exception Control Register* on page 2-11

2.3 AArch64 register descriptions

This section describes the Cortex-A53 processor Advanced SIMD and Floating-point system registers. Table 2-2 on page 2-3 provides cross-references to individual registers.

2.3.1 Floating-point Control Register

The FPCR characteristics are:

Purpose Controls floating-point extension behavior.

Usage constraints The accessibility to the FPCR by exception level is:

EL0	EL1 (NS)	EL1 (S)	EL2	EL3 (SCR.NS = 1)	EL3 (SCR.NS = 0)
RW	RW	RW	RW	RW	RW

Configurations The named fields in this register map to the equivalent fields in the AArch32 FPSCR. See *Floating-Point Status and Control Register* on page 2-15.

Attributes FPCR is a 32-bit register.

Figure 2-1 shows the FPCR bit assignments.

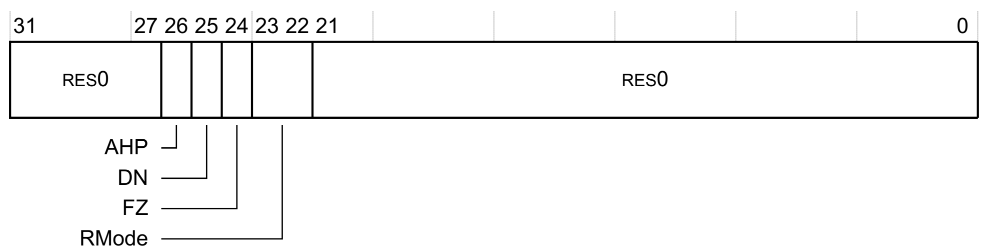

Figure 2-1 FPCR bit assignments

Table 2-3 shows the FPCR bit assignments.

Table 2-3 FPCR bit assignments

Bits	Name	Function
[31:27]	-	Reserved, RES0.
[26]	AHP	Alternative half-precision control bit. The possible values are: `0` IEEE half-precision format selected. `1` Alternative half-precision format selected.
[25]	DN	Default NaN mode control bit. The possible values are: `0` NaN operands propagate through to the output of a floating-point operation. `1` Any operation involving one or more NaNs returns the Default NaN.

Table 2-3 FPCR bit assignments (continued)

Bits	Name	Function
[24]	FZ	Flush-to-zero mode control bit. The possible values are: `0` Flush-to-zero mode disabled. Behavior of the floating-point system is fully compliant with the IEEE 754 standard. `1` Flush-to-zero mode enabled.
[23:22]	RMode	Rounding Mode control field. The encoding of this field is: `0b00` *Round to Nearest* (RN) mode. `0b01` *Round towards Plus Infinity* (RP) mode. `0b10` *Round towards Minus Infinity* (RM) mode. `0b11` *Round towards Zero* (RZ) mode.
[21:0]	-	Reserved, RES0.

To access the FPCR:

```
MRS <Xt>, FPCR ; Read FPCR into Xt
MSR FPCR, <Xt> ; Write Xt to FPCR
```

Table 2-4 shows the register access encoding.

Table 2-4 FPCR access encoding

op0	op1	CRn	CRm	op2
11	011	0100	0100	000

2.3.2 Floating-point Status Register

The FPSR characteristics are:

Purpose Provides floating-point system status information.

Usage constraints This register is accessible as follows:

EL0	EL1 (NS)	EL1 (S)	EL2	EL3 (SCR.NS = 1)	EL3 (SCR.NS = 0)
RW	RW	RW	RW	RW	RW

Configurations The named fields in this register map to the equivalent fields in the AArch32 FPSCR. See *Floating-Point Status and Control Register* on page 2-15.

Attributes FPSR is a 32-bit register.

Note

AArch64 floating-point comparisons set flags in the PSTATE register instead.

Figure 2-2 on page 2-6 shows the FPSR bit assignments.

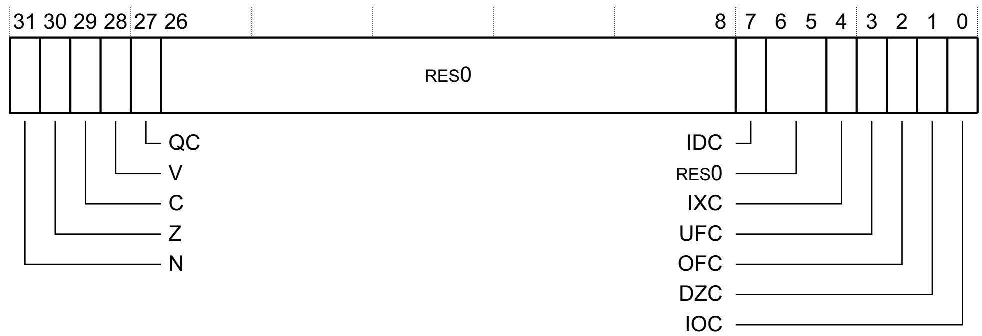

Figure 2-2 FPSR bit assignments

Table 2-5 shows the FPSR bit assignments.

Table 2-5 FPSR bit assignments

Bits	Name	Function
[31]	N	Negative condition flag for AArch32 floating-point comparison operations.
[30]	Z	Zero condition flag for AArch32 floating-point comparison operations.
[29]	C	Carry condition flag for AArch32 floating-point comparison operations.
[28]	V	Overflow condition flag for AArch32 floating-point comparison operations.
[27]	QC	Cumulative saturation bit. This bit is set to 1 to indicate that an Advanced SIMD integer operation has saturated since a 0 was last written to this bit.
[26:8]	-	Reserved, RES0.
[7]	IDC	Input Denormal cumulative exception bit. This bit is set to 1 to indicate that the Input Denormal exception has occurred since 0 was last written to this bit.
[6:5]	-	Reserved, RES0.
[4]	IXC	Inexact cumulative exception bit. This bit is set to 1 to indicate that the Inexact exception has occurred since 0 was last written to this bit.
[3]	UFC	Underflow cumulative exception bit. This bit is set to 1 to indicate that the Underflow exception has occurred since 0 was last written to this bit.
[2]	OFC	Overflow cumulative exception bit. This bit is set to 1 to indicate that the Overflow exception has occurred since 0 was last written to this bit.
[1]	DZC	Division by Zero cumulative exception bit. This bit is set to 1 to indicate that the Division by Zero exception has occurred since 0 was last written to this bit.
[0]	IOC	Invalid Operation cumulative exception bit. This bit is set to 1 to indicate that the Invalid Operation exception has occurred since 0 was last written to this bit.

To access the FPSR:

```
MRS <Xt>, FPSR; Read FPSR into Xt
MSR FPSR, <Xt>; Write Xt to FPSR
```

Table 2-6 shows the register access encoding.

Table 2-6 FPSR access encoding

op0	op1	CRn	CRm	op2
11	011	0100	0100	001

2.3.3 Media and Floating-point Feature Register 0

The MVFR0_EL1 characteristics are:

Purpose Describes the features provided by the Advanced SIMD and Floating-point Extension.

Usage constraints This register is accessible as follows:

EL0	EL1 (NS)	EL1 (S)	EL2	EL3 (SCR.NS = 1)	EL3 (SCR.NS = 0)
-	RO	RO	RO	RO	RO

Configurations MVFR0_EL1 is architecturally mapped to AArch32 register MVFR0. See *Media and Floating-point Feature Register 0* on page 2-17.

Attributes MVFR0_EL1 is a 32-bit register.

Figure 2-3 shows the MVFR0_EL1 bit assignments.

31 28	27 24	23 20	19 16	15 12	11 8	7 4	3 0
FPRound	FPShVec	FPSqrt	FPDivide	FPTrap	FPDP	FPSP	SIMDReg

Figure 2-3 MVFR0_EL1 bit assignments

Table 2-7 shows the MVFR0_EL1 bit assignments.

Table 2-7 MVFR0_EL1 bit assignments

Bits	Name	Function
[31:28]	FPRound	Indicates the rounding modes supported by the floating-point hardware: `0x1` All rounding modes supported.
[27:24]	FPShVec	Indicates the hardware support for floating-point short vectors: `0x0` Not supported.
[23:20]	FPSqrt	Indicates the hardware support for floating-point square root operations: `0x1` Supported.
[19:16]	FPDivide	Indicates the hardware support for floating-point divide operations: `0x1` Supported.
[15:12]	FPTrap	Indicates whether the floating-point hardware implementation supports exception trapping: `0x0` Not supported.

Table 2-7 MVFR0_EL1 bit assignments (continued)

Bits	Name	Function
[11:8]	FPDP	Indicates the hardware support for floating-point double-precision operations: `0x2` Supported, VFPv3 or greater. See the *ARM® Architecture Reference Manual, ARMv8* for more information.
[7:4]	FPSP	Indicates the hardware support for floating-point single-precision operations: `0x2` Supported, VFPv3 or greater. See the *ARM® Architecture Reference Manual, ARMv8* for more information.
[3:0]	SIMDReg	Indicates support for the Advanced SIMD register bank: `0x2` Supported, 32 x 64-bit registers supported. See the *ARM® Architecture Reference Manual, ARMv8* for more information.

To access the MVFR0_EL1:

```
MRS <Xt>, MVFR0_EL1 ; Read MVFR0_EL1 into Xt
```

Table 2-8 shows the register access encoding.

Table 2-8 MVFR0_EL1 access encoding

op0	op1	CRn	CRm	op2
11	000	0000	0011	000

2.3.4 Media and Floating-point Feature Register 1

The MVFR1_EL1 characteristics are:

Purpose Describes the features provided by the Advanced SIMD and Floating-point extensions.

Usage constraints This register is accessible as follows:

EL0	EL1 (NS)	EL1 (S)	EL2	EL3 (SCR.NS = 1)	EL3 (SCR.NS = 0)
-	RO	RO	RO	RO	RO

Configurations MVFR1_EL1 is architecturally mapped to AArch32 register MVFR1. See *Media and Floating-point Feature Register 1* on page 2-18.

Attributes MVFR1_EL1 is a 32-bit register.

Figure 2-4 shows the MVFR1_EL1 bit assignments.

31 28	27 24	23 20	19 16	15 12	11 8	7 4	3 0
SIMDFMAC	FPHP	SIMDHP	SIMDSP	SIMDInt	SIMDLS	FPDNaN	FPFtZ

Figure 2-4 MVFR1_EL1 bit assignments

Table 2-9 shows the MVFR1_EL1 bit assignments.

Table 2-9 MVFR1_EL1 bit assignments

Bits	Name	Function
[31:28]	SIMDFMAC	Indicates whether the Advanced SIMD and Floating-point Extension supports fused multiply accumulate operations: `0x1` Implemented.
[27:24]	FPHP	Indicates whether the Advanced SIMD and Floating-point Extension supports half-precision floating-point conversion instructions: `0x2` Instructions to convert between half-precision and single-precision, and between half-precision and double-precision, implemented.
[23:20]	SIMDHP	Indicates whether the Advanced SIMD and Floating-point extension supports half-precision floating-point conversion operations: `0x1` Implemented.
[19:16]	SIMDSP	Indicates whether the Advanced SIMD and Floating-point extension supports single-precision floating-point operations: `0x1` Implemented.
[15:12]	SIMDInt	Indicates whether the Advanced SIMD and Floating-point extension supports integer operations: `0x1` Implemented.
[11:8]	SIMDLS	Indicates whether the Advanced SIMD and Floating-point extension supports load/store instructions: `0x1` Implemented.
[7:4]	FPDNaN	Indicates whether the floating-point hardware implementation supports only the Default NaN mode: `0x1` Hardware supports propagation of NaN values.
[3:0]	FPFtZ	Indicates whether the floating-point hardware implementation supports only the Flush-to-Zero mode of operation: `0x1` Hardware supports full denormalized number arithmetic.

To access the MVFR1_EL1:

```
MRS <Xt>, MVFR1_EL1 ; Read MVFR1_EL1 into Xt
```

Table 2-10 shows the register access encoding.

Table 2-10 MVFR1_EL1 access encoding

op0	op1	CRn	CRm	op2
11	000	0000	0011	001

2.3.5 Media and Floating-point Feature Register 2

The MVFR2_EL1 characteristics are:

Purpose Describes the features provided by the Advanced SIMD and Floating-point extensions.

Usage constraints This register is accessible as follows:

EL0	EL1 (NS)	EL1 (S)	EL2	EL3 (SCR.NS = 1)	EL3 (SCR.NS = 0)
-	RO	RO	RO	RO	RO

Configurations MVFR2_EL1 is architecturally mapped to AArch32 register MVFR2. See *Media and Floating-point Feature Register 2* on page 2-20.

Attributes MVFR2_EL1 is a 32-bit register.

Figure 2-5 shows the MVFR2_EL1 bit assignments.

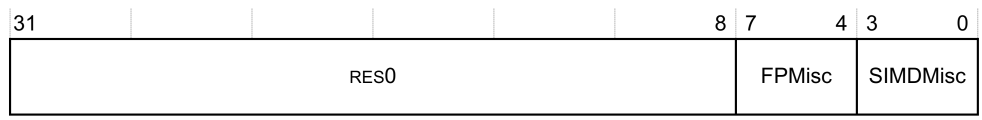

Figure 2-5 MVFR2_EL1 bit assignments

Table 2-11 shows the MVFR2_EL1 bit assignments.

Table 2-11 MVFR2_EL1 bit assignments

Bits	Name	Function
[31:8]	-	Reserved, RES0.
[7:4]	FPMisc	Indicates support for miscellaneous VFP features. `0x4` Supports: • Floating-point selection. • Floating-point Conversion to Integer with Directed Rounding modes. • Floating-point Round to Integral Floating-point. • Floating-point MaxNum and MinNum.
[3:0]	SIMDMisc	Indicates support for miscellaneous Advanced SIMD features. `0x3` Supports: • Floating-point Conversion to Integer with Directed Rounding modes. • Floating-point Round to Integral Floating-point. • Floating-point MaxNum and MinNum.

To access the MVFR2_EL1:

```
MRS <Xt>, MVFR2_EL1 ; Read MVFR2_EL1 into Xt
```

Table 2-12 shows the register access encoding.

Table 2-12 MVFR2_EL1 access encoding

op0	op1	CRn	CRm	op2
11	000	0000	0011	010

2.3.6 Floating-point Exception Control Register

The FPEXC32_EL2 characteristics are:

Purpose Provides access to the AArch32 register FPEXC from AArch64 state only. Its value has no effect on execution in AArch64 state.

Usage constraints This register is accessible as follows:

EL0	EL1 (NS)	EL1 (S)	EL2	EL3 (SCR.NS = 1)	EL3 (SCR.NS = 0)
-	-	-	RW	RW	RW

Configurations FPEXC32_EL2 is architecturally mapped to AArch32 register FPEXC. See *Floating-Point Exception Control register* on page 2-21.

Attributes FPEXC32_EL2 is a 32-bit register.

Figure 2-6 shows the FPEXC32_EL2 bit assignments.

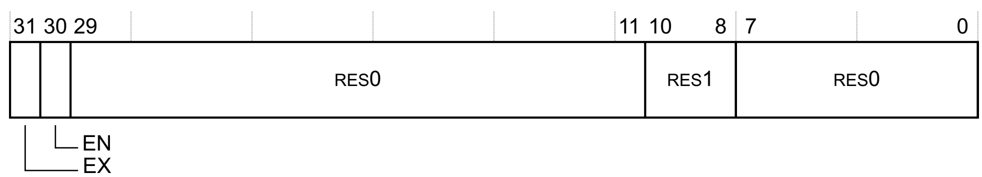

Figure 2-6 FPEXC32_EL2 bit assignments

Table 2-13 shows the FPEXC32_EL2 Register bit assignments.

Table 2-13 FPEXC32_EL2 bit assignments

Bits	Name	Function
[31]	EX	Exception bit. **RES0** The Cortex-A53 processor implementation does not generate asynchronous VFP exceptions.
[30]	EN	Enable bit. A global enable for the Advanced SIMD and VFP extensions: `0` The Advanced SIMD and VFP extensions are disabled. This is the reset value. `1` The Advanced SIMD and VFP extensions are enabled and operate normally. This bit applies only to AArch32 execution, and only when EL1 is not AArch64.
[29:11]	-	Reserved, RES0.
[10:8]	-	Reserved, RES1.
[7:0]	-	Reserved, RES0.

To access the FPEXC_EL2:

```
MRS <Xt>, FPEXC32_EL2 ; Read FPEXC32_EL2 into Xt
MSR FPEXC32_EL2, <Xt> ; Write Xt to FPEXC32_EL2
```

See also *Accessing the feature identification registers* on page 2-2.

Table 2-14 shows the register access encoding.

Table 2-14 FPEXC_EL2 access encoding

op0	op1	CRn	CRm	op2
11	100	0101	0011	000

2.4 AArch32 register summary

Table 2-15 gives a summary of the Cortex-A53 processor Advanced SIMD and Floating-point system registers in the AArch32 execution state.

Table 2-15 AArch32 Advanced SIMD and Floating-point system registers

Name	Type	Reset	Description
FPSID	RO	`0x41034033`	See *Floating-Point System ID Register* on page 2-14
FPSCR	RW	`0x00000000`	See *Floating-Point Status and Control Register* on page 2-15
MVFR0	RO	`0x10110222`	See *Media and Floating-point Feature Register 0* on page 2-17
MVFR1	RO	`0x12111111`	See *Media and Floating-point Feature Register 1* on page 2-18
MVFR2	RW	`0x00000043`	See *Media and Floating-point Feature Register 2* on page 2-20
FPEXC	RW	`0x00000700`	See *Floating-Point Exception Control register* on page 2-21

Note

The *Floating-Point Instruction Registers*, FPINST and FPINST2 are not implemented, and any attempt to access them is UNPREDICTABLE.

See the *ARM® Architecture Reference Manual, ARMv8* for information on permitted accesses to the Advanced SIMD and Floating-point system registers.

2.5 AArch32 register descriptions

This section describes the Cortex-A53 processor Advanced SIMD and Floating-point system registers. Table 2-15 on page 2-13 provides cross-references to individual registers.

2.5.1 Floating-Point System ID Register

The FPSID characteristics are:

Purpose Provides top-level information about the floating-point implementation.

Usage constraints This register is accessible as follows:

EL0 (NS)	EL0 (S)	EL1 (NS)	EL1 (S)	EL2	EL3 (SCR.NS = 1)	EL3 (SCR.NS = 0)
-	-	Config	RO	Config	Config	RO

Configurations Access to this register depends on the values of CPACR.{cp10,cp11}, NSACR.{cp10,cp11}, and HCPTR.{TCP10,TCP11}. For details of which field values permit access at specific exception levels, see the *ARM® Architecture Reference Manual, ARMv8*.

This register largely duplicates information held in the MIDR. ARM deprecates use of it.

Attributes FPSID is a 32-bit register.

Figure 2-7 shows the FPSID bit assignments.

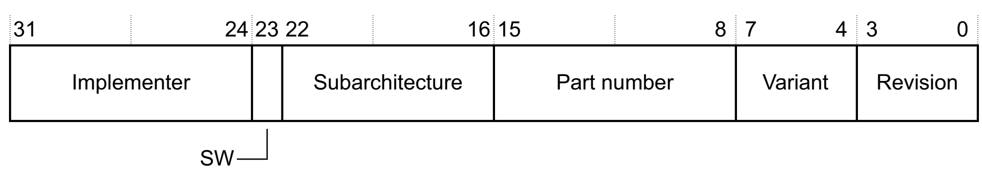

Figure 2-7 FPSID bit assignments

Table 2-16 shows the FPSID bit assignments.

Table 2-16 FPSID bit assignments

Bits	Name	Function
[31:24]	Implementer	Indicates the implementer: `0x41` ARM Limited.
[23]	SW	Software bit. This bit indicates that a system provides only software emulation of the floating-point instructions: `0` The system includes hardware support for floating-point operations.
[22:16]	Subarchitecture	Subarchitecture version number: `0x03` VFPv3 architecture, or later, with no subarchitecture. The entire floating-point implementation is in hardware, and requires no software support code. The MVFR0, MVFR1 and MVFR2 registers indicate the VFP architecture version.

Table 2-16 FPSID bit assignments (continued)

Bits	Name	Function
[15:8]	Part number	Indicates the part number for the floating-point implementation: `0x40` Cortex-A53 processor.
[7:4]	Variant	Indicates the variant number: `0x3` Cortex-A53 processor.
[3:0]	Revision	Indicates the revision number for the floating-point implementation: `0x3` r0p3.

To access the FPSID:

```
VMRS <Rt>, FPSID ; Read FPSID into Rt
```

2.5.2 Floating-Point Status and Control Register

The FPSCR characteristics are:

Purpose Provides floating-point system status information and control.

Usage constraints This register is accessible as follows:

EL0 (NS)	EL0 (S)	EL1 (NS)	EL1 (S)	EL2	EL3 (SCR.NS = 1)	EL3 (SCR.NS = 0)
Config	RW	Config	RW	Config	Config	RW

Access to this register depends on the values of CPACR.{cp10,cp11}, NSACR.{cp10,cp11}, HCPTR.{TCP10,TCP11} and FPEXC.EN. For details of which values of these fields allow access at which exception levels, see the *ARM® Architecture Reference Manual, ARMv8*.

Configurations There is one copy of this register that is used in both Secure and Nonsecure states.

The named fields in this register map to the equivalent fields in the AArch64 FPCR and FPSR. See *Floating-point Control Register* on page 2-4 and *Floating-point Status Register* on page 2-5.

Attributes FPSCR is a 32-bit register.

Figure 2-8 shows the FPSCR bit assignments.

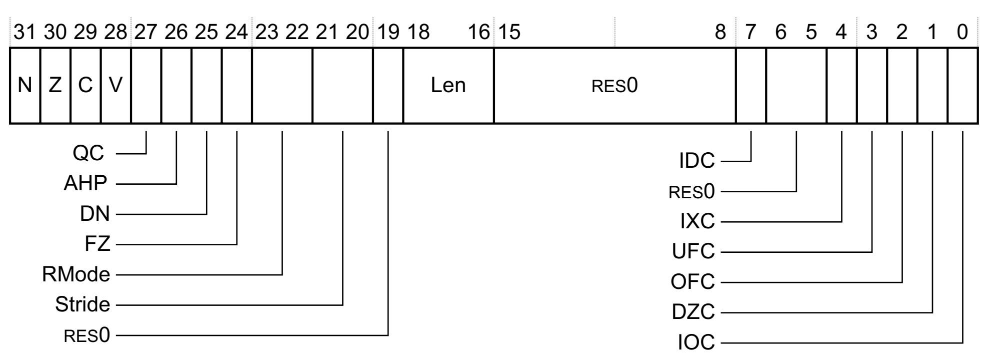

Figure 2-8 FPSCR bit assignments

Table 2-17 shows the FPSCR bit assignments.

Table 2-17 FPSCR bit assignments

Bits	Field	Function
[31]	N	Floating-point Negative condition code flag. Set to 1 if a floating-point comparison operation produces a less than result.
[30]	Z	Floating-point Zero condition code flag. Set to 1 if a floating-point comparison operation produces an equal result.
[29]	C	Floating-point Carry condition code flag. Set to 1 if a floating-point comparison operation produces an equal, greater than, or unordered result.
[28]	V	Floating-point Overflow condition code flag. Set to 1 if a floating-point comparison operation produces an unordered result.
[27]	QC	Cumulative saturation bit. This bit is set to 1 to indicate that an Advanced SIMD integer operation has saturated after 0 was last written to this bit.
[26]	AHP	Alternative Half-Precision control bit: `0` IEEE half-precision format selected. `1` Alternative half-precision format selected.
[25]	DN	Default NaN mode control bit: `0` NaN operands propagate through to the output of a floating-point operation. `1` Any operation involving one or more NaNs returns the Default NaN. The value of this bit only controls floating-point arithmetic. AArch32 Advanced SIMD arithmetic always uses the Default NaN setting, regardless of the value of the DN bit.
[24]	FZ	Flush-to-zero mode control bit: `0` Flush-to-zero mode disabled. Behavior of the floating-point system is fully compliant with the IEEE 754 standard. `1` Flush-to-zero mode enable. The value of this bit only controls floating-point arithmetic. AArch32 Advanced SIMD arithmetic always uses the Flush-to-zero setting, regardless of the value of the FZ bit.
[23:22]	RMode	Rounding Mode control field: `0b00` *Round to Nearest* (RN) mode. `0b01` *Round towards Plus Infinity* (RP) mode. `0b10` *Round towards Minus Infinity* (RM) mode. `0b11` *Round towards Zero* (RZ) mode. The specified rounding mode is used by almost all floating-point instructions. AArch32 Advanced SIMD arithmetic always uses the Round to Nearest setting, regardless of the value of the RMode bits.
[21:20]	Stride	RES0.
[19]	-	Reserved, RES0.
[18:16]	Len	RES0.
[15:8]	-	Reserved, RES0.
[7]	IDC	Input Denormal cumulative exception bit. This bit is set to 1 to indicate that the Input Denormal exception has occurred since 0 was last written to this bit.
[6:5]	-	Reserved, RES0.

Table 2-17 FPSCR bit assignments (continued)

Bits	Field	Function
[4]	IXC	Inexact cumulative exception bit. This bit is set to 1 to indicate that the Inexact exception has occurred since 0 was last written to this bit.
[3]	UFC	Underflow cumulative exception bit. This bit is set to 1 to indicate that the Underflow exception has occurred since 0 was last written to this bit.
[2]	OFC	Overflow cumulative exception bit. This bit is set to 1 to indicate that the Overflow exception has occurred since 0 was last written to this bit.
[1]	DZC	Division by Zero cumulative exception bit. This bit is set to 1 to indicate that the Division by Zero exception has occurred since 0 was last written to this bit.
[0]	IOC	Invalid Operation cumulative exception bit. This bit is set to 1 to indicate that the Invalid Operation exception has occurred since 0 was last written to this bit.

To access the FPSCR:

```
VMRS <Rt>, FPSCR ; Read FPSCR into Rt
VMSR FPSCR, <Rt> ; Write Rt to FPSCR
```

Note

The Cortex-A53 processor implementation does not support the deprecated VFP short vector feature. Attempts to execute VFP data-processing instructions result in an Undefined Instruction exception.

2.5.3 Media and Floating-point Feature Register 0

The MVFR0 characteristics are:

Purpose Describes the features provided by the Advanced SIMD and Floating-point Extension.

Usage constraints This register is accessible as follows:

EL0 (NS)	EL0 (S)	EL1 (NS)	EL1 (S)	EL2	EL3 (SCR.NS = 1)	EL3 (SCR.NS = 0)
-	-	Config	RO	Config	Config	RO

Access to this register depends on the values of CPACR.{cp10,cp11}, NSACR.{cp10,cp11}, HCPTR.{TCP10,TCP11}, and FPEXC.EN. For details of which values of these fields allow access at which exception levels, see the *ARM® Architecture Reference Manual, ARMv8*.

Must be interpreted with MVFR1 and MVFR2.See *Media and Floating-point Feature Register 1* on page 2-18 and *Media and Floating-point Feature Register 2* on page 2-20.

Configurations MVFR0 is architecturally mapped to AArch64 register MVFR0_EL1. See *Media and Floating-point Feature Register 0* on page 2-7.

There is one copy of this register that is used in both Secure and Non-secure states.

Attributes MVFR0 is a 32-bit register.

Figure 2-9 shows the MVFR0 bit assignments.

31 28	27 24	23 20	19 16	15 12	11 8	7 4	3 0
FPRound	FPShVec	FPSqrt	FPDivide	FPTrap	FPDP	FPSP	SIMDReg

Figure 2-9 MVFR0 bit assignments

Table 2-18 shows the MVFR0 bit assignments.

Table 2-18 MVFR0 bit assignments

Bits	Name	Function
[31:28]	FPRound	Indicates the rounding modes supported by the floating-point hardware: `0x1` All rounding modes supported.
[27:24]	FPShVec	Indicates the hardware support for floating-point short vectors: `0x0` Not supported.
[23:20]	FPSqrt	Indicates the hardware support for floating-point square root operations: `0x1` Supported.
[19:16]	FPDivide	Indicates the hardware support for floating-point divide operations: `0x1` Supported.
[15:12]	FPTrap	Indicates whether the floating-point hardware implementation supports exception trapping: `0x0` Not supported.
[11:8]	FPDP	Indicates the hardware support for floating-point double-precision operations: `0x2` Supported, VFPv3 or greater. See the *ARM® Architecture Reference Manual, ARMv8* for more information.
[7:4]	FPSP	Indicates the hardware support for floating-point single-precision operations: `0x2` Supported, VFPv3 or greater. See the *ARM® Architecture Reference Manual, ARMv8* for more information.
[3:0]	SIMDReg	Indicates support for the Advanced SIMD register bank: `0x2` Supported, 32 x 64-bit registers supported. See the *ARM® Architecture Reference Manual, ARMv8* for more information.

To access the MVFR0:

```
VMRS <Rt>, MVFR0 ; Read MVFR0 into Rt
```

2.5.4 Media and Floating-point Feature Register 1

The MVFR1 characteristics are:

Purpose Describes the features provided by the Advanced SIMD and Floating-point extensions.

Usage constraints This register is accessible as follows:

EL0 (NS)	EL0 (S)	EL1 (NS)	EL1 (S)	EL2	EL3 (SCR.NS = 1)	EL3 (SCR.NS = 0)
-	-	Config	RO	Config	Config	RO

Access to this register depends on the values of CPACR.{cp10,cp11}, NSACR.{cp10,cp11}, HCPTR.{TCP10,TCP11}, and FPEXC.EN. For details of which values of these fields allow access at which exception levels, see the *ARM® Architecture Reference Manual, ARMv8*.

Must be interpreted with MVFR0 and MVFR2.See *Media and Floating-point Feature Register 0* on page 2-17 and *Media and Floating-point Feature Register 2* on page 2-20

Configurations MVFR1 is architecturally mapped to AArch64 register MVFR1_EL1. See *Media and Floating-point Feature Register 1* on page 2-8.

There is one copy of this register that is used in both Secure and Non-secure states.

Attributes MVFR1 is a 32-bit register.

Figure 2-10 shows the MVFR1 bit assignments.

31 28	27 24	23 20	19 16	15 12	11 8	7 4	3 0
SIMDFMAC	FPHP	SIMDHP	SIMDSP	SIMDInt	SIMDLS	FPDNaN	FPFtZ

Figure 2-10 MVFR1 bit assignments

Table 2-19 shows the MVFR1 bit assignments.

Table 2-19 MVFR1 bit assignments

Bits	Name	Function
[31:28]	SIMDFMAC	Indicates whether the Advanced SIMD and Floating-point Extension supports fused multiply accumulate operations: `0x1` Implemented.
[27:24]	FPHP	Indicates whether the Advanced SIMD and Floating-point Extension supports half-precision floating-point conversion instructions: `0x2` Instructions to convert between half-precision and single-precision, and between half-precision and double-precision, implemented.
[23:20]	SIMDHP	Indicates whether the Advanced SIMD and Floating-point extension supports half-precision floating-point conversion operations: `0x1` Implemented.
[19:16]	SIMDSP	Indicates whether the Advanced SIMD and Floating-point extension supports single-precision floating-point operations: `0x1` Implemented.
[15:12]	SIMDInt	Indicates whether the Advanced SIMD and Floating-point extension supports integer operations: `0x1` Implemented.
[11:8]	SIMDLS	Indicates whether the Advanced SIMD and Floating-point extension supports load/store instructions: `0x1` Implemented.
[7:4]	FPDNaN	Indicates whether the floating-point hardware implementation supports only the Default NaN mode: `0x1` Hardware supports propagation of NaN values.
[3:0]	FPFtZ	Indicates whether the floating-point hardware implementation supports only the Flush-to-Zero mode of operation: `0x1` Hardware supports full denormalized number arithmetic.

To access the MVFR1:

```
VMRS <Rt>, MVFR1 ; Read MVFR1 into Rt
```

2.5.5 Media and Floating-point Feature Register 2

The MVFR2 characteristics are:

Purpose Describes the features provided by the Advanced SIMD and Floating-point extensions.

Usage constraints This register is accessible as follows:

EL0 (NS)	EL0 (S)	EL1 (NS)	EL1 (S)	EL2	EL3 (SCR.NS = 1)	EL3 (SCR.NS = 0)
-	-	Config	RO	Config	Config	RO

Access to this register depends on the values of CPACR.{cp10,cp11}, NSACR.{cp10,cp11}, HCPTR.{TCP10,TCP11}, and FPEXC.EN. For details of which values of these fields allow access at which exception levels, see the *ARM® Architecture Reference Manual, ARMv8*.

Must be interpreted with MVFR0 and MVFR1. See *Media and Floating-point Feature Register 0* on page 2-17 and *Media and Floating-point Feature Register 1* on page 2-18

Configurations MVFR2 is architecturally mapped to AArch64 register MVFR2_EL1. See *Media and Floating-point Feature Register 2* on page 2-9.

There is one copy of this register that is used in both Secure and Non-secure states.

Attributes MVFR2 is a 32-bit register.

Figure 2-11 shows the MVFR2 bit assignments.

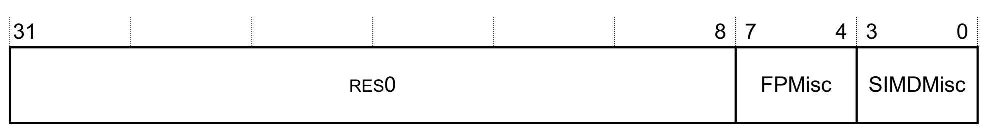

Figure 2-11 MVFR2 bit assignments

Table 2-20 shows the MVFR2 bit assignments.

Table 2-20 MVFR2 bit assignments

Bits	Name	Function
[31:8]	-	Reserved, RES0.
[7:4]	FPMisc	Indicates support for miscellaneous VFP features. `0x4` Supports: • Floating-point selection. • Floating-point Conversion to Integer with Directed Rounding modes. • Floating-point Round to Integral Floating-point. • Floating-point MaxNum and MinNum.
[3:0]	SIMDMisc	Indicates support for miscellaneous Advanced SIMD features. `0x3` Supports: • Floating-point Conversion to Integer with Directed Rounding modes. • Floating-point Round to Integral Floating-point. • Floating-point MaxNum and MinNum.

To access the MVFR2:

```
VMRS <Rt>, MVFR2 ; Read MVFR2 into Rt
```

2.5.6 Floating-Point Exception Control register

The FPEXC characteristics are:

Purpose Provides a global enable for the Advanced SIMD and Floating-point extension, and indicates how the state of this extension is recorded.

Usage constraints This register is accessible as follows:

EL0 (NS)	EL0 (S)	EL1 (NS)	EL1 (S)	EL2	EL3 (SCR.NS = 1)	EL3 (SCR.NS = 0)
-	-	Config	RW	Config	Config	RW

Access to this register depends on the values of CPACR.{cp10,cp11}, NSACR.{cp10,cp11}, and HCPTR.{TCP10,TCP11}. For details of which values of these fields allow access at which exception levels, see the *ARM® Architecture Reference Manual, ARMv8*.

Configurations FPEXC is architecturally mapped to AArch64 register FPEXC32_EL2. See *Floating-point Exception Control Register* on page 2-11.

There is one copy of this register that is used in both Secure and Non-secure states.

Attributes FPEXC is a 32-bit register.

Figure 2-12 on page 2-22 shows the FPEXC bit assignments.

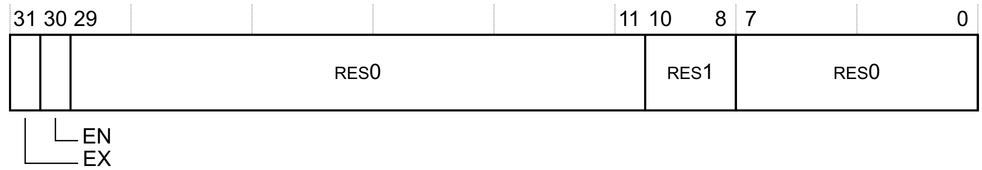

Figure 2-12 FPEXC bit assignments

Table 2-21 shows the FPEXC Register bit assignments.

Table 2-21 FPEXC bit assignments

Bits	Name	Function
[31]	EX	Exception bit. The Cortex-A53 processor implementation does not generate asynchronous VFP exceptions, therefore this bit is RES0.
[30]	EN	Enable bit. A global enable for the Advanced SIMD and VFP extensions: `0` The Advanced SIMD and VFP extensions are disabled. `1` The Advanced SIMD and VFP extensions are enabled and operate normally. The EN bit is cleared at reset. This bit applies only to AArch32 execution, and only when EL1 is not AArch64.
[29:11]	-	Reserved, RES0.
[10:8]	-	Reserved, RES1.
[7:0]	-	Reserved, RES0.

To access the FPEXC register:

```
VMRS <Rt>, FPEXC ; Read FPEXC into Rt
```

See also *Accessing the feature identification registers* on page 2-2.

Appendix A
Revisions

This appendix describes the technical changes between released issues of this book.

Table A-1 Issue A

Change	Location	Affects
First Release	-	-

Table A-2 Differences between Issue A and Issue B

Change	Location	Affects
FPSID register updated	Table 2-15 on page 2-13 Table 2-16 on page 2-14	r0p1

Table A-3 Differences between Issue B and Issue C

Change	Location	Affects
FPSID register updated	Table 2-15 on page 2-13 Table 2-16 on page 2-14	r0p2

Table A-4 Differences between Issue C and Issue D

Change	Location	Affects
There are no technical changes.	-	-

Table A-5 Differences between Issue C and Issue D

Change	Location	Affects
FPSID register updated	Table 2-15 on page 2-13 Table 2-16 on page 2-14	r0p3