ARM® Cortex®-A5 Floating-Point Unit

Revision: r0p1

Technical Reference Manual

ARM 100302_0001_00_en

ARM® Cortex®-A5 Floating-Point Unit

Technical Reference Manual

Release Information

Document History

Issue	Date	Confidentiality	Change
A	18 December 2009	Non-Confidential	First release for r0p0
B	30 September 2010	Non-Confidential	First release for r0p1
0001-00	15 April 2015	Non-Confidential	Source content converted to DITA. Document number changed to 100302. No technical changes have been made to the contents of the book.

Non-Confidential Proprietary Notice

ARM Limited. Company 02557590 registered in England.

110 Fulbourn Road, Cambridge, England CB1 9NJ.

LES-PRE-20349

Confidentiality Status

This document is Non-Confidential. The right to use, copy and disclose this document may be subject to license restrictions in accordance with the terms of the agreement entered into by ARM and the party that ARM delivered this document to.

Unrestricted Access is an ARM internal classification.

Product Status

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

Web Address

http://www.arm.com

Contents

ARM® Cortex®-A5 Floating-Point Unit Technical Reference Manual

Preface

This preface introduces the *ARM® Cortex®-A5 Floating-Point Unit Technical Reference Manual*.

It contains the following:

About this book

This book is for the Cortex-A5 *Floating-Point Unit* (FPU)

Product revision status

The r*m*p*n* identifier indicates the revision status of the product described in this book, for example, r*1*p*2*, 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 verification engineers who are designing a *System-on-Chip* (SoC) device that uses the FPU. The book describes the external functionality of the FPU

Using this book

This book is organized into the following chapters:

Chapter 1 Introduction
This chapter introduces the Cortex-A5 FPU.

Chapter 2 Programmers Model
This chapter describes mplementation-specific features of the Cortex-A5 FPU that are useful to programmers.

Appendix A Revisions
This appendix describes 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 the *ARM Glossary* for more information.

Typographic conventions

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.

`mono`space
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.

Timing diagrams

The following figure explains the components used in timing diagrams. Variations, when they occur, have clear labels. You must not assume any timing information that is not explicit in the diagrams.

Shaded bus and signal areas are undefined, so the bus or signal can assume any value within the shaded area at that time. The actual level is unimportant and does not affect normal operation.

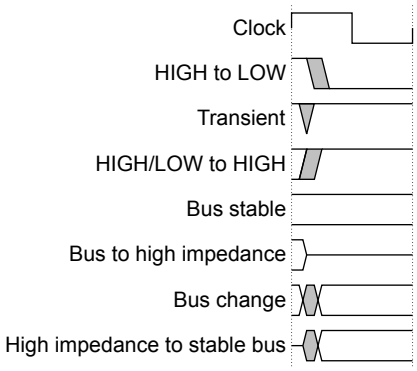

Figure 1 Key to timing diagram conventions

Signals

The signal conventions are:

Signal level
: The level of an asserted signal depends on whether the signal is active-HIGH or active-LOW. Asserted means:
- HIGH for active-HIGH signals.
- LOW for active-LOW signals.

Lower-case n
: At the start or end of a signal name denotes an active-LOW signal.

Additional reading

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

ARM publications

- *Cortex®-A5 Technical Reference Manual* (ARM DDI 0433)
- *Cortex®-A5 MPCore Technical Reference Manual* (ARM DDI 0434)
- *Cortex®-A5 Supplementary Datasheet* (ARM DDI 0448)
- *ARM® Cortex®-A5 NEON Media Processing Engine Technical Reference Manual* (ARM 100304)
- *Cortex®-A5 Configuration and Sign-Off Guide* (ARM DII 0210)
- *ARM® Cortex®-A5 Integration Manual* (ARM 100312)
- *ARM® Architecture Reference Manual, ARMv7-A and ARMv7-R edition* (ARM DDI 0406)
- *CoreSight™ ETM-A5 Technical Reference Manual* (ARM DDI 0435)
- *CoreSight™ ETM-A5 Configuration and Sign-Off Guide* (ARM DII 0212)
- *CoreSight™ ETM-A5 Integration Manual* (ARM DIT 0002)
- *CoreSight™ Design Kit for Cortex®-A5 Integration Manual* (ARM DIT 0003)
- *CoreSight™ Embedded Trace Macrocell v3.5 Architecture Specification* (ARM IHI 0014)
- *PrimeCell Level 2 Cache Controller (PL310) Technical Reference Manual* (ARM DDI 0246)
- *AMBA® AXI Protocol v1.0 Specification* (ARM IHI 0022)
- *ARM® Generic Interrupt Controller Architecture Specification* (ARM IHI 0048)
- *RealView ICE User Guide* (ARM DUI 0155)
- *Intelligent Energy Controller Technical Overview* (ARM DTO 0005)
- *CoreSight™ Architecture Specification* (ARM IHI 0029)
- *CoreSight™ Technology System Design Guide* (ARM DGI 0012)
- *ARM® Cortex®-A5 FPU Release Note* (AT551-DC-06002).

Other publications

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

Feedback

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 100302_0001_00_en.
- The page number(s) to which your comments refer.
- 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 Cortex-A5 FPU.

It contains the following sections:

1.1 About the Cortex-A5 FPU

The Cortex-A5 FPU is a VFPv4-D16 implementation of the ARMv7 floating-point architecture. It provides low-cost high performance floating-point computation.

The FPU supports all addressing modes and operations that are described in the *ARM® Architecture Reference Manual, ARMv7-A and ARMv7-R edition*.

The FPU features are:

- Support for single-precision and double-precision floating-point formats.
- Support for conversion between half-precision and single-precision.
- Support for *Fused Multiply Accumulate* (FMA) operations.
- Normalized and denormalized data are all handled in hardware.
- Trapless operation enabling fast execution.

This section contains the following subsections:

- *1.1.1 VFPv4 architecture hardware support* on page 1-11.

1.1.1 VFPv4 architecture hardware support

The Cortex-A5 FPU hardware supports single and double-precision add, subtract, multiply, divide, multiply and accumulate, fused multiply accumulate, and square root operations as described in the ARM VFPv4 architecture.

It provides conversions between 16-bit, 32-bit, and 64-bit floating-point formats and ARM integer word formats, with special operations to perform conversions in round-towards-zero mode for high-level language support.

ARMv7 deprecates the use of VFP vector mode. The Cortex-A5 FPU hardware does not support VFP vector operations. The Cortex-A5 FPU provides high-speed VFP operation without support code. However, if an application requires VFP vector operation, then it must use support code. See the *ARM® Architecture Reference Manual, ARMv7-A and ARMv7-R edition* for information on VFP vector operation support.

Note

This manual gives information specific to the Cortex-A5 FPU implementation of the ARM VFPv4 extension. See the *ARM® Architecture Reference Manual, ARMv7-A and ARMv7-R edition* for full instruction set and usage details.

1.2 Applications

The Cortex-A5 FPU provides floating-point computation suitable for a wide spectrum of applications.

Example applications are:

- Personal digital assistants and smartphones for graphics, voice compression and decompression, user interfaces, Java interpretation, and *Just In Time* (JIT) compilation.
- Games machines for three-dimensional graphics and digital audio.
- Printers and *MultiFunction Peripheral* (MFP) controllers for high-definition color rendering.
- Set-top boxes for digital audio and digital video, and three-dimensional user interfaces.
- Automotive applications for engine management and power train computations.

1.3 Product revisions

You can find information about the differences in functionality between product revisions.

r0p0
: First release.

r0p0-r0p1
: No functional changes.

r0p1-r0p1
: Source content conversion to DITA. No technical changes.

Chapter 2
Programmers Model

This chapter describes mplementation-specific features of the Cortex-A5 FPU that are useful to programmers.

It contains the following sections:

2.1 About the programmers model

The Cortex-A5 FPU implementation of the VFPv4 floating-point architecture, VFPv4-D16, with version 2 of the Common VFP subarchitecture supports the following functionality.

- All scalar operations are implemented entirely in hardware, with support for all combinations of rounding modes, flush-to-zero, and Default NaN modes.
- Vector operations are not supported. Any attempt to execute a vector operation results in a synchronous bounce, with the FPEXC.DEX bit set to 1.
- The Cortex-A5 FPU never generates an asynchronous VFP exception.

In addition, it provides information on initializing the Cortex-A5 FPU ready for application code execution.

Two access types are supported:

RW Read and write.

RO Read only.

This section contains the following subsections:
- *2.1.1 VFP feature identification registers* on page 2-15.
- *2.1.2 Enabling VFP support* on page 2-15.
- *2.1.3 Instructions for FPU* on page 2-16.

2.1.1 VFP feature identification registers

The Cortex-A5 FPU implements the ARMv7 VFP extension.

Software can identify this extension and the features it provides, using the feature identification registers. This extension is in the coprocessor space for coprocessors CP10 and CP11. You can access the registers using the `VMRS` and `VMSR` instructions, for example:

```
VMRS <Rd>, FPSID ; Read Floating-Point System ID Register
VMRS <Rd>, MVFR1 ; Read Media and VFP Feature Register 1
VMSR FPSCR, <Rt> ; Write Floating-Point System Control Register
```

In addition, there are registers and coprocessor access control registers.

Related references

2.2 VFP register access on page 2-17.

2.1.2 Enabling VFP support

From reset, the VFP extension is disabled. Any attempt to execute a VFP instruction results in an Undefined Instruction exception being taken.

To enable software to access VFP features, ensure that:

- Access to CP10 and CP11 is enabled for the appropriate privilege level.
- If Non-secure access to the VFP features is required, the access flags for CP10 and CP11 in the NSACR must be set to 1.

In addition, software must set the FPEXC.EN bit to 1 to enable most VFP operations.

When a VFP operation is disabled because FPEXC.EN is 0, all VFP instructions are treated as undefined instructions except for execution of the following in privileged modes:
- A VMSR to the FPEXC or FPSID register
- A VMRS from the FPEXC, FPSID, MVFR0, or MVFR1 registers.

Related references

2.2 VFP register access on page 2-17.

2.4.5 Floating-point Exception Register on page 2-23.

2.1.3 Instructions for FPU

You can use the FPU in two different states.

Using the FPU in Secure state only

To use the FPU in Secure state only, define the CPACR and FPEXC registers to enable the FPU.

Procedure

1. Set the CPACR for access to CP10 and CP11 (the FPU coprocessors):

```
LDR r0, =(0xF << 20)
MCR p15, 0, r0, c1, c0, 2
```

2. Set the FPEXC.EN bit to enable the FPU:

```
MOV r3, #0x40000000
VMSR FPEXC, r3
```

Using the FPU in Secure state and Non-secure state

To use the FPU in Secure state and Non-secure state, first define the NSACR and then define the CPACR and FPEXC registers to enable the FPU.

Procedure

1. Set bits [11:10] of the NSACR for access to CP10 and CP11 from both Secure and Non-secure states:

```
MRC p15, 0, r0, c1, c1, 2
ORR r0, r0, #2_11<<10 ; enable fpu
MCR p15, 0, r0, c1, c1, 2
```

2. Set the CPACR for access to CP10 and CP11:

```
LDR r0, =(0xF << 20)
MCR p15, 0, r0, c1, c0, 2
```

3. Set the FPEXC.EN bit to enable the FPU:

```
MOV r3, #0x40000000
VMSR FPEXC, r3
```

——— **Note** ———

Operation is UNPREDICTABLE if you configure the CPACR such that CP10 and CP11 do not have identical access permissions.

———————————

2.2 VFP register access

System control coprocessor registers, which are accessed through CP15, determine access to VFP registers.

- CRn is the register number within CP15.
- Op1 is the Opcode_1 value for the register.
- CRm is the operational register.
- Op2 is the Opcode_2 value for the register.

Table 2-1 Coprocessor Access Control registers

CRn	Op1	CRm	Op2	Name	Description
c1	0	c0	2	CPACR	See the *Cortex®-A5 Technical Reference Manual*
c1	0	c1	2	NSACR	See the *Cortex®-A5 Technical Reference Manual*

2.3 Register summary

All FPU system registers are 32-bit wide. Reserved register addresses are UNPREDICTABLE.

The following table shows the Cortex-A5 FPU system registers.

Table 2-2 Cortex-A5 FPU system registers

Name	Type	Reset	Description
FPSID	RO	0x41023051	*2.4.1 Floating-point System ID Register* on page 2-19
FPSCR	RW	0x00000000	*2.4.2 Floating-point Status and Control Register* on page 2-20
MVFR0	RO	0x10110221	*2.4.3 Media and VFP Feature Register 0* on page 2-21
MVFR1	RO	0x11000011	*2.4.4 Media and VFP Feature Register 1* on page 2-22
FPEXC	RW	0x00000000	*2.4.5 Floating-point Exception Register* on page 2-23

Note

The FPINST and FPINST2 registers are not implemented, and any attempt to access them is UNPREDICTABLE.

The following table shows the processor modes for accessing the Cortex-A5 FPU system registers.

Table 2-3 Accessing Cortex-A5 FPU system registers

Register	Privileged access		User access	
	FPEXC.EN=0	FPEXC.EN=1	FPEXC.EN=0	FPEXC.EN=1
FPSID	Permitted	Permitted	Not permitted	Not permitted
FPSCR	Not permitted	Permitted	Not permitted	Permitted
MVFR0, MVFR1	Permitted	Permitted	Not permitted	Not permitted
FPEXC	Permitted	Permitted	Not permitted	Not permitted

2.4 Register descriptions

This section describes the FPU system registers.

This section contains the following subsections:

2.4.1 Floating-point System ID Register

The FPSID Register characteristics are:

Purpose

Provides information about the VFP implementation.

Usage constraints

This register is:

- Only accessible in the Non-secure state if the CP10 and CP11 bits in the NSACR are set to 1. See *2.2 VFP register access* on page 2-17.
- Only accessible in privileged modes, and only if access to coprocessors CP10 and CP11 is enabled in the CPACR and FPEXC.EN is set. See *2.2 VFP register access* on page 2-17.

Configurations

Available in all configurations.

Attributes

See *2.3 Register summary* on page 2-18.

The following figure shows the FPSID Register bit assignments.

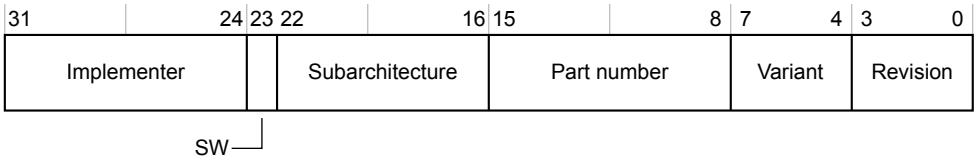

Figure 2-1 FPSID Register bit assignments

The following table shows the FPSID Register bit assignments.

Table 2-4 FPSID Register bit assignments

Bits	Name	Function
[31:24]	Implementer	Denotes ARM. Value is `0x41`.
[23]	SW	Hardware implementation with no software emulation. Value is `0`.
[22:16]	Subarchitecture	VFPv3 or greater with v2 subarchitecture. Value is `2`.
[15:8]	Part number	Cortex-A. Value is `0x30`.
[7:4]	Variant	Cortex-A5. Value is `5`.
[3:0]	Revision	Revision. Value is `1`.

You can access the FPSID Register with the following `VMRS` instruction:

```
VMRS <Rd>, FPSID ; Read Floating-Point System ID Register
```

Related references

2.2 VFP register access on page 2-17.

2.3 Register summary on page 2-18.

2.4.2 Floating-point Status and Control Register

The FPSCR characteristics are:

Purpose

Provides User-level control of the FPU.

Usage constraints

This register is:

- Only accessible in the Non-secure state if the CP10 and CP11 bits in the NSACR are set to 1. See *2.2 VFP register access* on page 2-17.
- Accessible in all modes depending on the setting of bits [23:20] of the CPACR and FPEXC.EN. See *2.2 VFP register access* on page 2-17.

Configurations

Available in all configurations.

Attributes

See *2.3 Register summary* on page 2-18.

The following figure shows the FPSCR bit assignments.

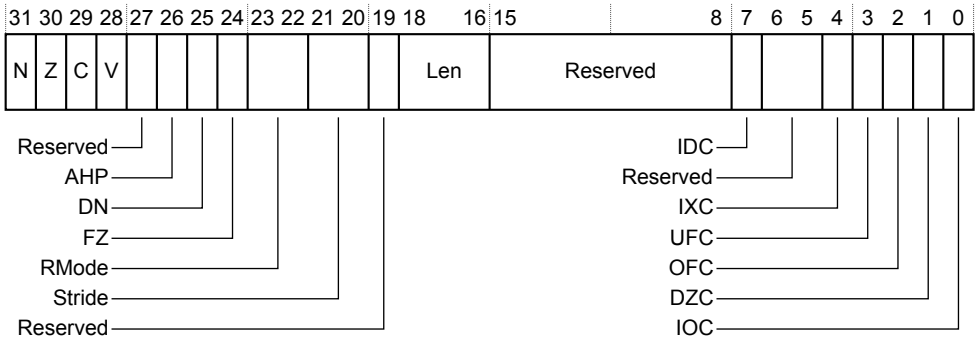

Figure 2-2 FPSCR bit assignments

The following table shows the FPSCR bit assignments.

Table 2-5 FPSCR bit assignments

Bits	Name	Function
[31]	N	Set to 1 if a comparison operation produces a less than result.
[30]	Z	Set to 1 if a comparison operation produces an equal result.
[29]	C	Set to 1 if a comparison operation produces an equal, greater than, or unordered result.
[28]	V	Set to 1 if a comparison operation produces an unordered result.
[27]	Reserved	UNK/SBZP.
[26]	AHP	Alternative Half-Precision control bit: b0 = IEEE half-precision format selected. b1 = Alternative half-precision format selected.

Table 2-5 FPSCR bit assignments (continued)

Bits	Name	Function
[25]	DN	Default NaN mode control bit: b0 = NaN operands propagate through to the output of a floating-point operation. b1 = Any operation involving one or more NaNs returns the Default NaN.
[24]	FZ	Flush-to-zero mode control bit: b0 = Flush-to-zero mode disabled. Behavior of the floating-point system is fully compliant with the IEEE 754 standard. b1 = Flush-to-zero mode enabled.
[23:22]	RMode	Rounding mode control field: b00 = *Round to nearest* (RN) mode b01 = *Round towards plus infinity* (RP) mode b10 = *Round towards minus infinity* (RM) mode b11 = *Round towards zero* (RZ) mode.
[21:20]	Stride	Stride control used for backwards compatibility with short vector values. See the *ARM Architecture Reference Manual, ARMv7-A and ARMv7-R edition*.
[19]	Reserved	UNK/SBZP.
[18:16]	Len	Vector length, used for backwards compatibility with short vector values. See the *ARM Architecture Reference Manual, ARMv7-A and ARMv7-R edition*.
[15:8]	Reserved	UNK/SBZP.
[7]	IDC	Input Denormal cumulative exception flag.
[6:5]	Reserved	UNK/SBZP.
[4]	IXC	Inexact cumulative exception flag.
[3]	UFC	Underflow cumulative exception flag.
[2]	OFC	Overflow cumulative exception flag.
[1]	DZC	Division by Zero cumulative exception flag.
[0]	IOC	Invalid Operation cumulative exception flag.

You can access the FPSCR with the following `VMSR` instructions:

```
VMRS <Rd>, FPSCR ; Read Floating-Point Status and Control Register
VMSR FPSCR, <Rt> ; Write Floating-Point Status and Control Register
```

Related references

2.2 VFP register access on page 2-17.
2.3 Register summary on page 2-18.

2.4.3 Media and VFP Feature Register 0

The MVFR0 characteristics are:

Purpose

Together with MVFR1, describes the features that the FPU provides.

Usage constraints

This register is:

- Only accessible in the Non-secure state if the CP10 and CP11 bits in the NSACR are set to 1. See *2.2 VFP register access* on page 2-17.
- Only accessible in privileged modes, and only if access to coprocessors CP10 and CP11 is enabled in the CPACR and FPEXC.EN is set. See *2.2 VFP register access* on page 2-17.

Configurations

Available in all configurations.

Attributes

See *2.3 Register summary* on page 2-18.

The following figure shows the MVFR0 bit assignments.

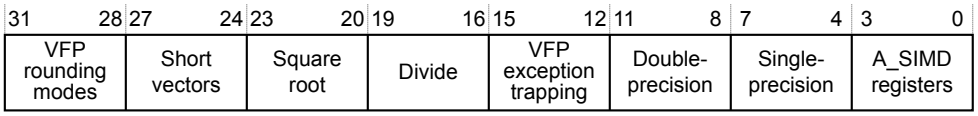

Figure 2-3 MVFR0 bit assignments

The following table shows the MVFR0 bit assignments.

Table 2-6 MVFR0 bit assignments

Bits	Name	Function
[31:28]	VFP rounding modes	All VFP rounding modes supported
[27:24]	Short vectors	VFP short vectors not supported
[23:20]	Square root	VFP square root operation supported
[19:16]	Divide	VFP divide operation supported
[15:12]	VFP exception trapping	VFP exception trapping not supported
[11:8]	Double-precision	Double-precision operations supported
[7:4]	Single-precision	Single-precision operations supported
[3:0]	A_SIMD registers	Sixteen 64-bit registers supported

You can access the MVFR0 with the following `VMSR` instruction:

```
VMRS <Rd>, MVFR0 ; Read Media and VFP Feature Register 0
```

Related references

2.2 VFP register access on page 2-17.

2.3 Register summary on page 2-18.

2.4.4 Media and VFP Feature Register 1

The MVFR1 characteristics are:

Purpose

Together with MVFR0, describes the features that the FPU provides.

Usage constraints

This register is:

- Only accessible in the Non-secure state if the CP10 and CP11 bits in the NSACR are set to 1. See *2.2 VFP register access* on page 2-17.
- Only accessible in privileged modes, and only if access to coprocessors CP10 and CP11 is enabled in the CPACR and FPEXC.EN is set. See *2.2 VFP register access* on page 2-17.

Configurations
Available in all configurations.

Attributes
See *2.3 Register summary* on page 2-18.

The following figure shows the MVFR1 bit assignments.

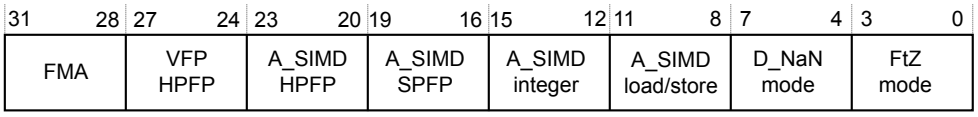

Figure 2-4 MVFR1 bit assignments

The following table shows the MVFR1 bit assignments.

Table 2-7 MVFR1 bit assignments

Bits	Name	Function
[31:28]	FMA	Fused Multiply Accumulate supported
[27:24]	VFP HPFP	VFP half-precision operations supported
[23:20]	A_SIMD HPFP	Advanced SIMD half-precision operations not supported
[19:16]	A_SIMD SPFP	Advanced SIMD single-precision operations not supported
[15:12]	A_SIMD integer	Advanced SIMD integer operations not supported
[11:8]	A_SIMD load/store	Advanced SIMD load/store instructions not supported
[7:4]	D_NaN mode	Propagation of NaN values supported for VFP
[3:0]	FtZ mode	Full denormal arithmetic operations supported for VFP

You can access the MVFR1 with the following `VMSR` instruction:

```
VMRS <Rd>, MVFR1 ; Read Media and VFP Feature Register 1
```

Related references

2.2 VFP register access on page 2-17.

2.3 Register summary on page 2-18.

2.4.5 Floating-point Exception Register

The FPEXC Register characteristics are:

Purpose
Provides global enable control of the VFP extension.

Usage constraints
This register is:

- Only accessible in the Non-secure state if the CP10 and CP11 bits in the NSACR are set to 1. See *2.2 VFP register access* on page 2-17.
- Only accessible in privileged modes, and only if access to coprocessors CP10 and CP11 is enabled in the CPACR. See *2.2 VFP register access* on page 2-17.

Configurations
Available in all configurations.

Attributes
See *2.3 Register summary* on page 2-18.

The following figure shows the FPEXC Register bit assignments.

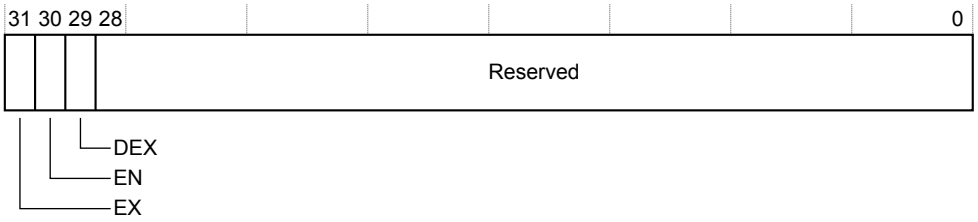

Figure 2-5 FPEXC Register bit assignments

The following table shows the FPEXC Register bit assignments.

Table 2-8 FPEXC Register bit assignments

Bits	Name	Function
[31]	EX	The Cortex-A5 FPU does not generate asynchronous VFP exceptions, therefore this bit is RAZ/WI.
[30]	EN	FPU enable bit: b0 = FPU disabled. b1 = FPU enabled. The EN bit is cleared to 0 at reset.
[29]	DEX	Defined synchronous instruction exceptional flag. The Cortex-A5 FPU sets this bit when generating a synchronous bounce because of an attempt to execute a vector operation. All other UNDEFINED Instruction exceptions clear this bit to zero. See the *ARM Architecture Reference Manual, ARMv7-A and ARMv7-R edition* for more information.
[28:0]	Reserved	RAZ/WI.

You can access the FPEXC Register with the following `VMSR` instructions:

```
VMRS <Rd>, FPEXC ; Read Floating-Point Status and Control Register
VMSR FPEXC, <Rt> ; Write Floating-Point Status and Control Register
```

Related references

2.2 VFP register access on page 2-17.
2.3 Register summary on page 2-18.

Appendix A
Revisions

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

It contains the following sections:

- *A.1 Revisions* on page Appx-A-26.

A.1 Revisions

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

Table A-1 Issue A

Change	Location	Affects
No changes, first release	-	-

Table A-2 Differences between issue A and issue B

Change	Location	Affects
FPSID reset value updated	*2.3 Register summary* on page 2-18	r0p1
Implementation revision value updated	*2.4.1 Floating-point System ID Register* on page 2-19	r0p1