ARM® Cortex®-A55 Core Cryptographic Extension

Revision: r1p0

Technical Reference Manual

ARM

ARM 100444_0100_00_en

ARM® Cortex®-A55 Core Cryptographic Extension

Technical Reference Manual

Release Information

Document History

Issue	Date	Confidentiality	Change
0000-00	30 September 2016	Confidential	First release for r0p0
0001-00	16 December 2016	Confidential	First release for r0p1
0100-00	22 June 2017	Non-Confidential	First release for r1p0

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®-A55 Core Cryptographic Extension Technical Reference Manual

Preface

This preface introduces the *ARM® Cortex®-A55 Core Cryptographic Extension Technical Reference Manual*.

It contains the following:

About this book

This document describes the optional cryptographic features of the Cortex-A55 core. It includes descriptions of the registers used by the Cryptographic Extension.

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 manual is written for system designers, system integrators, and programmers who are designing or programming a System-on-Chip (SoC) that uses the Cortex®-A55 core with the optional Cryptographic Extension.

Using this book

This book is organized into the following chapters:

Chapter 1 Functional description

This chapter describes the Cortex-A55 core Cryptographic Extension.

Chapter 2 Register descriptions

This chapter describes the Cryptographic Extension registers.

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.

Additional reading

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

ARM publications

- *ARM® Cortex®-A55 Core Technical Reference Manual* (ARM 100442)
- *ARM® Cortex®-A55 Core Configuration and Sign-off Guide* (ARM 100443)
- *ARM® Cortex®-A55 Core Integration Manual* (ARM 100445)
- *ARM® Cortex®-A55 Core Advanced SIMD and Floating-point Support Technical Reference Manual* (ARM 100446)
- *ARM® Architecture Reference Manual ARMv8, for ARMv8-A architecture profile* (ARM DDI 0487)

Other publications

- *Advanced Encryption Standard.* (FIPS 197, November 2001).
- *Secure Hash Standard (SHS)* (FIPS 180-4, March 2012).

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 *ARM Cortex-A55 Core Cryptographic Extension Technical Reference Manual*.
- The number ARM 100444_0100_00_en.
- If applicable, 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
Functional description

This chapter describes the Cortex-A55 core Cryptographic Extension.

It contains the following sections:

1.1 About the Cryptographic Extension

The Cortex-A55 core Cryptographic Extension supports the ARMv8 Cryptographic Extension.

The Cryptographic Extension adds new A64, A32, and T32 instructions to Advanced SIMD that accelerate *Advanced Encryption Standard* (AES) encryption and decryption. It also adds instructions to implement the *Secure Hash Algorithm* (SHA) functions SHA-1, SHA-224, and SHA-256.

Note

The optional Cryptographic Extension is not included in the base product. ARM supplies the Cryptographic Extension only under an additional license to the Cortex-A55 core and Advanced SIMD and floating-point support licenses.

1.2 Revisions

This section describes the differences in functionality between product revisions.

r0p0 First release.

r0p1 No differences in functionality.

r1p0 No differences in functionality.

Chapter 2
Register descriptions

This chapter describes the Cryptographic Extension registers.

It contains the following sections:

2.1 Identifying the cryptographic instructions implemented

Software can identify the cryptographic instructions that are implemented by reading three registers.

The three registers are:

- ID_AA64ISAR0_EL1 in the AArch64 execution state.
- ID_ISAR5_EL1 in the AArch64 execution state.
- ID_ISAR5 in the AArch32 execution state.

2.2 Disabling the Cryptographic Extension

To disable the Cryptographic Extension, assert the **CRYPTODISABLE** input signal that applies to all the Cortex-A55 cores present in a cluster. This signal is sampled only during reset of the cores.

When **CRYPTODISABLE** is asserted:

- Executing a cryptographic instruction results in an UNDEFINED exception.
- The ID registers described in *Table 2-1 Cryptographic Extension register summary* on page 2-15 indicate that the Cryptographic Extension is not implemented.

2.3 Register summary

The Cortex-A55 core has three instruction identification registers. Each register has a specific purpose, usage constraints, configurations, and attributes.

The following table lists the instruction identification registers for the Cortex-A55 core Cryptographic Extension.

Table 2-1 Cryptographic Extension register summary

Name	Execution state	Description
ID_AA64ISAR0_EL1	AArch64	See *2.4 ID_AA64ISAR0_EL1, AArch64 Instruction Set Attribute Register 0, EL1* on page 2-16.
ID_ISAR5_EL1	AArch64	See *2.5 ID_ISAR5_EL1, AArch32 Instruction Set Attribute Register 5, EL1* on page 2-18.
ID_ISAR5	AArch32	See *2.6 ID_ISAR5, Instruction Set Attribute Register 5* on page 2-20.

2.4 ID_AA64ISAR0_EL1, AArch64 Instruction Set Attribute Register 0, EL1

The ID_AA64ISAR0_EL1 provides information about the instructions implemented in AArch64 state, including the instructions provided by the Cryptographic Extension.

Bit field descriptions

ID_AA64ISAR0_EL1 is a 64-bit register.

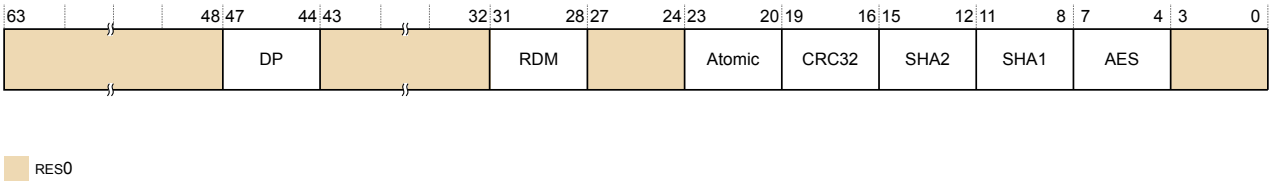

Figure 2-1 ID_AA64ISAR0_EL1 bit assignments

RES0, [63:48]

RES0 Reserved.

DP, [47:44]

Indicates whether Dot Product support instructions are implemented.

0x1 UDOT, SDOT instructions are implemented.

RES0, [43:32]

RES0 Reserved.

RDM, [31:28]

Indicates whether *Rounding Double Multiply* (RDM) instructions are implemented. The value is:

0x1 SQRDMLAH and SQRDMLSH instructions are implemented.

[27:24]

RES0 Reserved.

Atomic, [23:20]

Indicates whether atomic instructions are implemented. The value is:

0x2 LDADD, LDCLR, LDEOR, LDSET, LDSMAX, LDSMIN, LDUMAX, LDUMIN, CAS, CASP, and SWP instructions are implemented.

CRC32, [19:16]

Indicates whether CRC32 instructions are implemented. The value is:

0x1 CRC32 instructions are implemented.

SHA2, [15:12]

Indicates whether SHA2 instructions are implemented. The possible values are:

0x0 No SHA2 instructions are implemented. This is the value if the core implementation does not include the Cryptographic Extension.

0x1 SHA256H, SHA256H2, SHA256U0, and SHA256U1 are implemented. This is the value if the core implementation includes the Cryptographic Extension.

SHA1, [11:8]

Indicates whether SHA1 instructions are implemented. The possible values are:

0x0 No SHA1 instructions are implemented. This is the value if the core implementation does not include the Cryptographic Extension.

`0x1` `SHA1C`, `SHA1P`, `SHA1M`, `SHA1SU0`, and `SHA1SU1` are implemented. This is the value if the core implementation includes the Cryptographic Extension.

AES, [7:4]

Indicates whether AES instructions are implemented. The possible values are:

`0x0` No AES instructions implemented. This is the value if the core implementation does not include the Cryptographic Extension.

`0x2` `AESE`, `AESD`, `AESMC`, and `AESIMC` are implemented, plus `PMULL` and `PMULL2` instructions operating on 64-bit data. This is the value if the core implementation includes the Cryptographic Extension.

[3:0]

RES0 Reserved.

Configurations

ID_AA64ISAR0_EL1 is architecturally mapped to external register ID_AA64ISAR0.

Usage constraints

Accessing the ID_AA64ISAR0_EL1

To access the ID_AA64ISAR0_EL1:

```
MRS <Xt>, ID_AA64ISAR0_EL1 ; Read ID_AA64ISAR0_EL1 into Xt
```

Register access is encoded as follows:

Table 2-2 ID_AA64ISAR0_EL1 access encoding

op0	op1	CRn	CRm	op2
11	000	0000	0110	000

Accessibility

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

2.5 ID_ISAR5_EL1, AArch32 Instruction Set Attribute Register 5, EL1

The AArch64 register ID_ISAR5_EL1 provides information about the instructions implemented in AArch32 state, including the instructions provided by the optional Cryptographic Extension.

Bit field descriptions

ID_ISAR5_EL1 is a 32-bit register.

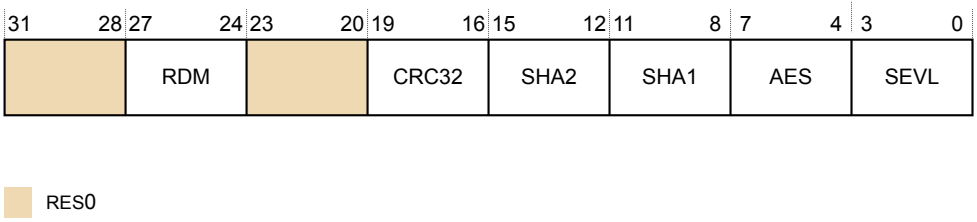

Figure 2-2 ID_ISAR5_EL1 bit assignments

[31:28]

RES0 Reserved.

RDM, [27:24]

Indicates whether RDM instructions are implemented. The value is:

0x1 SQRDMLAH and SQRDMLSH instructions are implemented.

[23:20]

RES0 Reserved.

CRC32, [19:16]

Indicates whether CRC32 instructions are implemented in AArch32 state. The value is:

0x1 CRC32 instructions are implemented.

SHA2, [15:12]

Indicates whether SHA2 instructions are implemented in AArch32 state. The possible values are:

0x0 Cryptographic Extension is not implemented or is disabled.

0x1 SHA256H, SHA256H2, SHA256SU0, and SHA256SU1 instructions are implemented.

SHA1, [11:8]

Indicates whether SHA1 instructions are implemented in AArch32 state. The possible values are:

0x0 Cryptographic Extension is not implemented or is disabled.

0x1 SHA1C, SHA1P, SHA1M, SHA1H, SHA1SU0, and SHA1SU1 instructions are implemented.

AES, [7:4]

Indicates whether AES instructions are implemented in AArch32 state. The possible values are:

0x0 Cryptographic Extension is not implemented or is disabled.

0x2 AESE, AESD, AESMC, and AESIMC are implemented, plus PMULL and PMULL2 instructions operating on 64-bit data.

SEVL, [3:0]

Indicates whether the SEVL instruction is implemented. The value is:

0x1 SEVL implemented to send event local.

Configurations

ID_ISAR5_EL1 is architecturally mapped to AArch32 register ID_ISAR5. See *2.6 ID_ISAR5, Instruction Set Attribute Register 5* on page 2-20.

Usage constraints

Accessing the ID_ISAR5_EL1

To access the ID_ISAR5_EL1:

```
MRS <Xt>, ID_ISAR5_EL1 ; Read ID_ISAR5_EL1 into Xt
```

Register access is encoded as follows:

Table 2-3 ID_ISAR5_EL1 access encoding

op0	op1	CRn	CRm	op2
11	000	0000	0010	101

Accessibility

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

2.6 ID_ISAR5, Instruction Set Attribute Register 5

The AArch32 register ID_ISAR5 provides information about the instructions implemented in AArch32 state, including the instructions provided by the optional Cryptographic Extension.

Bit field descriptions

ID_ISAR5 is a 32-bit register.

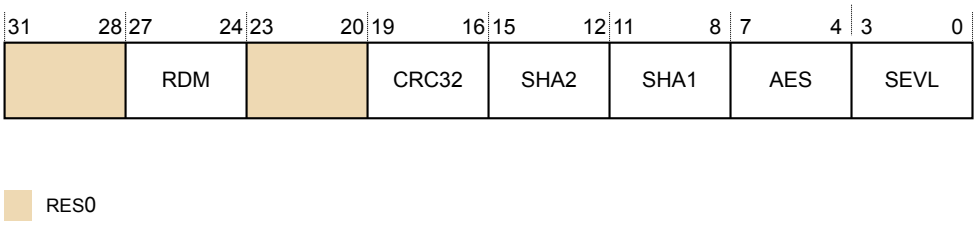

Figure 2-3 ID_ISAR5 bit assignments

[31:28]

RES0 Reserved.

RDM, [27:24]

Indicates whether RDM instructions are implemented. The value is:

0x1 SQRDMLAH and SQRDMLSH instructions are implemented.

[23:20]

RES0 Reserved.

CRC32, [19:16]

Indicates whether CRC32 instructions are implemented in AArch32 state. The value is:

0x1 CRC32 instructions are implemented.

SHA2, [15:12]

Indicates whether SHA2 instructions are implemented in AArch32 state. The possible values are:

0x0 Cryptographic extension is not implemented or is disabled.

0x1 SHA256H, SHA256H2, SHA256SU0, and SHA256SU1 instructions are implemented.

SHA1, [11:8]

Indicates whether SHA1 instructions are implemented in AArch32 state. The possible values are:

0x0 Cryptographic Extension is not implemented or is disabled.

0x1 SHA1C, SHA1P, SHA1M, SHA1H, SHA1SU0, and SHA1SU1 instructions are implemented.

AES, [7:4]

Indicates whether AES instructions are implemented in AArch32 state. The possible values are:

0x0 Cryptographic Extension is not implemented or is disabled.

0x2 AESE, AESD, AESMC and AESIMC, plus PMULL and PMULL2 instructions operating on 64-bit data.

SEVL, [3:0]

Indicates whether the SEVL instruction is implemented. The value is:

0x1 SEVL implemented to send event local.

Configurations

ID_ISAR5 is architecturally mapped to AArch64 register ID_ISAR5_EL1. See *2.5 ID_ISAR5_EL1, AArch32 Instruction Set Attribute Register 5, EL1* on page 2-18.

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

Usage constraints

Accessing the ID_ISAR5

To access ID_ISAR5:

```
MRC p15, 0, <Rt>, c0, c2, 5; Read ID_ISAR5 into Rt
```

This register is accessible as follows:

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

Appendix A
Revisions

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

It contains the following section:

- *A.1 Revisions* on page Appx-A-23.

A.1 Revisions

This section describes the technical changes between released issues of this document.

Table A-1 Issue 0000-00

Change	Location	Affects
First release	-	-

Table A-2 Differences between issue 0000-00 and issue 0001-00

Change	Location	Affects
Changed revision to r0p1	*1.2 Revisions* on page 1-11	r0p1

Table A-3 Differences between issue 0001-00 and issue 0100-00

Change	Location	Affects
Changed revision to r1p0	*1.2 Revisions* on page 1-11	r1p0
Updated product name	-	r1p0
Global terminology change from 'processor' to 'core' for the product	-	r1p0
Updated the ID_AA64ISAR0_EL1 register description	*2.4 ID_AA64ISAR0_EL1, AArch64 Instruction Set Attribute Register 0, EL1* on page 2-16	r1p0