# ARM® Cortex®-A53 MPCore Processor Cryptography Extension

**Revision: r0p4**

## Technical Reference Manual

ARM®


ARM DDI 0501F (ID041316)

# ARM Cortex-A53 MPCore Processor Cryptography 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 |
| 30 April 2014 | D | Confidential | Release for r0p3 |
| 29 July 2014 | E | Confidential | Release for r0p4 |
| 16 December 2015 | F | Non-Confidential | Second release for r0p4 |

### Proprietary Notice



### 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.

### 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 Cryptography Extension Technical Reference Manual

# Preface

This preface introduces the *ARM® Cortex®-A53 MPCore Cryptography Extension Technical Reference Manual*. It contains the following sections:

# About this book

This book is for the Cortex-A53 MPCore Cryptography 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 Cryptography Extension.

## Using this book

This book is organized into the following chapters:

**Chapter 1 *Introduction***

Read this for an introduction to the Cortex-A53 processor Cryptography Extension.

**Chapter 2 *Programmers Model***

Read this for a description of the Cortex-A53 processor Cryptography Extension programmers model.

**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:

| 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. |
| `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 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® Cortex®-A53 MPCore Processor Advanced SIMD and Floating-point Extension Technical Reference Manual* (ARM DDI 0502).
- *ARM® Architecture Reference Manual, ARMv8, for ARMv8-A architecture profile* (ARM DDI 0487).
- *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:

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

## 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 0501F.
- 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 describes the Cortex-A53 MPCore Cryptography Extension. It contains the following sections:

- *About the Cortex-A53 processor Cryptography Extension* on page 1-2.
- *Revisions* on page 1-3.

## 1.1 About the Cortex-A53 processor Cryptography Extension

The Cortex-A53 processor Cryptography Extension supports the ARMv8 Cryptography Extensions. The Cryptography Extensions add new A64, A32, and T32 instructions to Advanced SIMD that accelerate *Advanced Encryption Standard* (AES) encryption and decryption, and the *Secure Hash Algorithm* (SHA) functions SHA-1, SHA-224, and SHA-256.

**Note**

The optional Cryptography Extension is not included in the base product. ARM supplies the Cryptography Extension only under an additional licence to the Cortex-A53 processor and Advanced SIMD and Floating-point support licences.

## 1.2 Revisions

This section describes the differences in functionality between product revisions:

**r0p0** First release.

**r0p1** There are no functional changes in this revision.

**r0p2** There are no functional changes in this revision.

**r0p3** There are no functional changes in this revision.

**r0p4** There are no functional changes in this revision.

# Chapter 2
# **Programmers Model**

This chapter describes the programmers model. It contains the following sections:

## 2.1 About the programmers model

This section describes the registers of the Cortex-A53 processor Cryptography Extension and provides programming information. See the *ARM® Architecture Reference Manual, ARMv8* for more information.

This section describes:

- *Identifying the cryptography instructions implemented*.
- *Disabling the Cryptography Extension*.

### 2.1.1 Identifying the cryptography instructions implemented

Software can identify the cryptography instructions implemented by reading:

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

### 2.1.2 Disabling the Cryptography Extension

To disable the Cryptography Extension for each individual core, assert the corresponding bit of the **CRYPTODISABLE** input signal. This signal is only sampled during reset of the core.

When **CRYPTODISABLE** is asserted:

- Executing a cryptography instruction results in an UNDEFINED exception.
- The ID registers described in Table 2-1 on page 2-3 indicate that the Cryptography Extension is not implemented.

## 2.2 Register summary

Table 2-1 lists the instruction identification registers for the Cortex-A53 processor Cryptography Extension.

**Table 2-1 Cryptography extension register summary**

| Name | Execution state | Description |
|---|---|---|
| ID_AA64ISAR0_EL1 | AArch64 | See *AArch64 Instruction Set Attribute Register 0, EL1* on page 2-4. |
| ID_ISAR5 | AArch32 | See *Instruction Set Attribute Register 5* on page 2-6. |
| ID_ISAR5_EL1 | AArch64 | See *Instruction Set Attribute Register 5* on page 2-6. |

## 2.3 Register descriptions

This section describes the Cortex-A53 processor Cryptography Extension registers. Table 2-1 on page 2-3 provides cross references to individual registers.

### 2.3.1 AArch64 Instruction Set Attribute Register 0, EL1

The ID_AA64ISAR0_EL1 characteristics are:

**Purpose** Provides information about the optional cryptography instructions that the processor can support.

**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** ID_AA64ISAR0_EL1 is architecturally mapped to external register ID_AA64ISAR0.

**Attributes** ID_AA64ISAR0_EL1 is a 64-bit register.

Figure 2-1 shows the ID_AA64ISAR0_EL1 bit assignments.

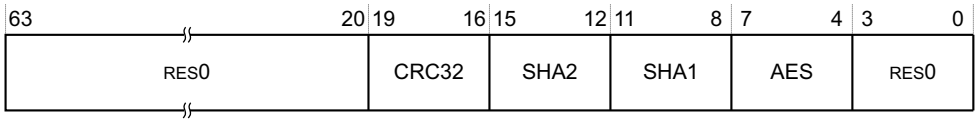


**Figure 2-1 ID_AA64ISAR0_EL1 bit assignments**

Table 2-2 shows the ID_AA64ISAR0_EL1 bit assignments.

**Table 2-2 ID_AA64ISAR0_EL1 bit assignments**

| Bits | Name | Function |
|---|---|---|
| [63:20] | - | Reserved, RES0. |
| [19:16] | CRC32 | `0x1` CRC32 instructions are implemented. |
| [15:12] | SHA2 | Indicates whether SHA2 instructions are implemented. The possible values are:<br>`0x0` No SHA2 instructions are implemented. This is the value if the implementation does not include the Cryptography Extension.<br>`0x1` `SHA256H`, `SHA256H2`, `SHA256U0`, and `SHA256U1` implemented. This is the value if the implementation includes the Cryptography Extension. |

**Table 2-2 ID_AA64ISAR0_EL1 bit assignments (continued)**

| Bits | Name | Function |
|---|---|---|
| [11:8] | SHA1 | Indicates whether SHA1 instructions are implemented. The possible values are:<br>`0x0` No SHA1 instructions implemented. This is the value if the implementation does not include the Cryptography Extension<br>`0x1` `SHA1C`, `SHA1P`, `SHA1M`, `SHA1SU0`, and `SHA1SU1` implemented. This is the value if the implementation includes the Cryptography Extension. |
| [7:4] | AES | Indicates whether AES instructions are implemented. The possible values are:<br>`0x0` No AES instructions implemented. This is the value if the implementation does not include the Cryptography Extension<br>`0x2` `AESE`, `AESD`, `AESMC`, and `AESIMC` implemented, plus `PMULL` and `PMULL2` instructions operating on 64-bit data. This is the value if the implementation includes the Cryptography Extension. |
| [3:0] | - | Reserved, RES0. |

To access the ID_AA64ISAR0_EL1:

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

ID_AA64ISAR0_EL1[31:0] can be accessed through the internal memory-mapped interface and the external debug interface, offset `0xD30`.

Register access is encoded as follows:

**Table 2-3 ID_AA64ISAR0_EL1 access encoding**

| op0 | op1 | CRn | CRm | op2 |
|---|---|---|---|---|
| 11 | 000 | 0000 | 0110 | 000 |

### 2.3.2 AArch32 Instruction Set Attribute Register 5

The ID_ISAR5_EL1 characteristics are:

**Purpose** Provides information about the instruction sets that the processor implements.

**Note**

The optional Cryptography Extension is not included in the base product of the processor. ARM requires licensees to have contractual rights to obtain the Cryptography 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** ID_ISAR5_EL1 is architecturally mapped to AArch32 register ID_ISAR5. See *Instruction Set Attribute Register 5* on page 2-6.

**Attributes** ID_ISAR5_EL1 is a 32-bit register.

Figure 2-2 on page 2-6 shows the ID_ISAR5_EL1 bit assignments.

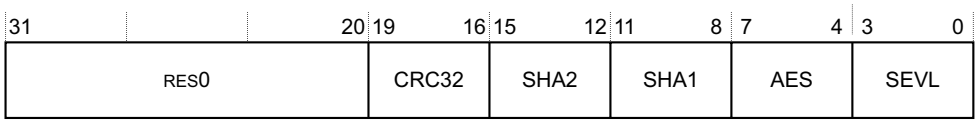


**Figure 2-2 ID_ISAR5_EL1 bit assignments**

Table 2-4 shows the ID_ISAR5_EL1 bit assignments.

**Table 2-4 ID_ISAR5_EL1 bit assignments**

| Bits | Name | Function |
|---|---|---|
| [31:20] | - | Reserved, RES0. |
| [19:16] | CRC32 | Indicates whether CRC32 instructions are implemented in AArch32 state. The value is:<br>0x1 CRC32 instructions are implemented. |
| [15:12] | SHA2 | Indicates whether SHA2 instructions are implemented in AArch32 state. The possible values are:<br>0x0 Cryptography Extensions are not implemented or are disabled.<br>0x1 SHA256H, SHA256H2, SHA256SU0, and SHA256SU1 instructions are implemented. |
| [11:8] | SHA1 | Indicates whether SHA1 instructions are implemented in AArch32 state. The possible values are:<br>0x0 Cryptography Extensions are not implemented or are disabled.<br>0x1 SHA1C, SHA1P, SHA1M, SHA1H, SHA1SU0, and SHA1SU1 instructions are implemented. |
| [7:4] | AES | Indicates whether AES instructions are implemented in AArch32 state. The possible values are:<br>0x0 Cryptography Extensions are not implemented or are disabled.<br>0x2 AESE, AESD, AESMC, and AESIMC are implemented, plus PMULL and PMULL2 instructions operating on 64-bit data. |
| [3:0] | SEVL | Indicates whether the SEVL instruction is implemented. The value is:<br>0x1 SEVL implemented to send event local. |

To access the ID_ISAR5_EL1:

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

Register access is encoded as follows:

**Table 2-5 ID_ISAR5_EL1 access encoding**

| op0 | op1 | CRn | CRm | op2 |
|---|---|---|---|---|
| 11 | 000 | 0000 | 0010 | 101 |

### 2.3.3 Instruction Set Attribute Register 5

The ID_ISAR5 characteristics are:

**Purpose** Provides information about the instruction sets implemented by the processor in AArch32.

**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) |
|---|---|---|---|---|---|---|
| - | - | RO | RO | RO | RO | RO |

The ID_ISAR5 must be interpreted with ID_ISAR0, ID_ISAR1, ID_ISAR2, ID_ISAR3, and ID_ISAR4.

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

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

**Attributes** ID_ISAR5 is a 32-bit register.

.

Figure 2-3 shows the ID_ISAR5 bit assignments.

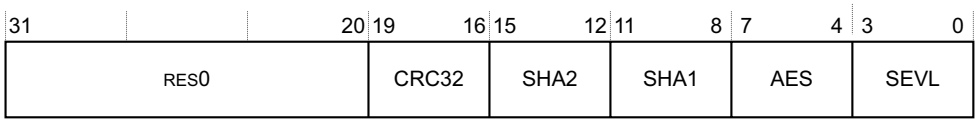


**Figure 2-3 ID_ISAR5 bit assignments**

Table 2-6 shows the ID_ISAR5 bit assignments.

**Table 2-6 ID_ISAR5 bit assignments**

| Bits | Name | Function |
|---|---|---|
| [31:20] | - | Reserved, RES0. |
| [19:16] | CRC32 | Indicates whether CRC32 instructions are implemented in AArch32 state. The value is:<br>0x1 CRC32 instructions are implemented. |
| [15:12] | SHA2 | Indicates whether SHA2 instructions are implemented in AArch32 state. The possible values are:<br>0x0 Cryptographic extensions are not implemented or are disabled.<br>0x1 SHA256H, SHA256H2, SHA256SU0, and SHA256SU1 instructions are implemented. |
| [11:8] | SHA1 | Indicates whether SHA1 instructions are implemented in AArch32 state. The possible values are:<br>0x0 Cryptographic extensions are not implemented or are disabled.<br>0x1 SHA1C, SHA1P, SHA1M, SHA1H, SHA1SU0, and SHA1SU1 instructions are implemented. |
| [7:4] | AES | Indicates whether AES instructions are implemented in AArch32 state. The possible values are:<br>0x0 Cryptographic extensions are not implemented or are disabled.<br>0x2 AESE, AESD, AESMC and AESIMC, plus PMULL and PMULL2 instructions operating on 64-bit data. |
| [3:0] | SEVL | Indicates whether the SEVL instruction is implemented. The value is:<br>0x1 SEVL implemented to send event local. |

To access ID_ISAR5:

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

# Appendix A
# **Revisions**

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

**Table A-1 Issue A**

| Change | Location | Affects |
|---|---|---|
| First revision | - | - |

**Table A-2 Differences between Issue A and Issue B**

| Change | Location | Affects |
|---|---|---|
| There are no technical changes between these released issues. | - | - |

**Table A-3 Differences between Issue B and Issue C**

| Change | Location | Affects |
|---|---|---|
| There are no technical changes between these released issues. | - | - |

**Table A-4 Differences between Issue C and Issue D**

| Change | Location | Affects |
|---|---|---|
| There are no technical changes between these released issues. | - | - |

**Table A-5 Differences between Issue D and Issue E**

| Change | Location | Affects |
|---|---|---|
| There are no technical changes between these released issues. | - | - |

**Table A-6 Differences between Issue E and Issue F**

| Change | Location | Affects |
|---|---|---|
| Document confidentiality changed. | Throughout | Issue F |