Arm® Cortex®-X3 Core Cryptographic Extension

Revision: r1p1

Technical Reference Manual

Non-Confidential

Issue 04

101602_0101_04_en

Arm® Cortex®-X3 Core Cryptographic Extension

Technical Reference Manual

Release Information

Document history

Issue	Date	Confidentiality	Change
0000-01	11 November 2020	Confidential	First Beta release for r0p0
0000-02	19 February 2021	Confidential	First limited access release for r0p0
0100-03	22 July 2021	Confidential	First early access release for r1p0
0101-04	28 June 2022	Non-Confidential	First early access release for r1p1

Proprietary Notice

This document is protected by copyright and other related rights and the practice or implementation of the information contained in this document may be protected by one or more patents or pending patent applications. No part of this document may be reproduced in any form by any means without the express prior written permission of Arm. No license, express or implied, by estoppel or otherwise to any intellectual property rights is granted by this document unless specifically stated.

Your access to the information in this document is conditional upon your acceptance that you will not use or permit others to use the information for the purposes of determining whether implementations infringe any third party patents.

THIS DOCUMENT IS PROVIDED “AS IS”. ARM PROVIDES NO REPRESENTATIONS AND NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY, SATISFACTORY QUALITY, NON-INFRINGEMENT OR FITNESS FOR A PARTICULAR PURPOSE WITH RESPECT TO THE DOCUMENT. For the avoidance of doubt, Arm makes no representation with respect to, has undertaken no analysis to identify or understand the scope and content of, third party patents, copyrights, trade secrets, or other rights.

This document may include technical inaccuracies or typographical errors.

TO THE EXTENT NOT PROHIBITED BY LAW, IN NO EVENT WILL ARM BE LIABLE FOR ANY DAMAGES, INCLUDING WITHOUT LIMITATION ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, PUNITIVE, OR CONSEQUENTIAL DAMAGES, HOWEVER CAUSED AND

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.

Feedback

Arm welcomes feedback on this product and its documentation. To provide feedback on the product, create a ticket on https://support.developer.arm.com.

To provide feedback on the document, fill the following survey: https://developer.arm.com/documentation-feedback-survey.

Inclusive language commitment

Arm values inclusive communities. Arm recognizes that we and our industry have used language that can be offensive. Arm strives to lead the industry and create change.

This document includes language that can be offensive. We will replace this language in a future issue of this document.

To report offensive language in this document, email terms@arm.com.

Contents

1. Introduction

1.1 Product revision status

The `r`*`x`*`p`*`y`* identifier indicates the revision status of the product described in this manual, for example, `r`*`1`*`p`*`2`*, where:

`r`*`x`*	Identifies the major revision of the product, for example, r1.
`p`*`y`*	Identifies the minor revision or modification status of the product, for example, p2.

1.2 Intended audience

This manual is for system designers, system integrators, and programmers who are designing or programming a *System-on-Chip* (SoC) that uses the Cortex®-X3 core with the optional Cryptographic Extension.

1.3 Conventions

The following subsections describe conventions used in Arm documents.

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: developer.arm.com/glossary.

Typographic conventions

Convention	Use
italic	Citations.
bold	Interface elements, such as menu names. Signal names. Terms in descriptive lists, where appropriate.
`monospace`	Text that you can enter at the keyboard, such as commands, file and program names, and source code.
`monospace bold`	Language keywords when used outside example code.

Convention	Use
`monospace underline`	A permitted abbreviation for a command or option. You can enter the underlined text instead of the full command or option name.
`<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	Terms that have specific technical meanings as defined in the *Arm® Glossary*. For example, **IMPLEMENTATION DEFINED**, **IMPLEMENTATION SPECIFIC**, **UNKNOWN**, and **UNPREDICTABLE**.

Caution

Recommendations. Not following these recommendations might lead to system failure or damage.

Warning

Requirements for the system. Not following these requirements might result in system failure or damage.

Danger

Requirements for the system. Not following these requirements will result in system failure or damage.

Note

An important piece of information that needs your attention.

Tip

A useful tip that might make it easier, better or faster to perform a task.

Remember

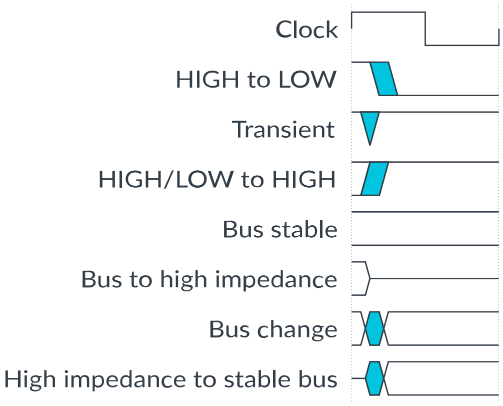
A reminder of something important that relates to the information you are reading.

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 1-1: 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.

Lowercase n

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

1.4 Additional reading

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

Table 1-2: Arm publications

Document Name	Document ID	Licensee only
Arm® Cortex®-X3 Core Technical Reference Manual	101593	Yes
Arm® Cortex®-X3 Core Configuration and Integration Manual	101594	Yes

Document Name	Document ID	Licensee only
Arm® Architecture Reference Manual Armv8, for A-profile architecture	DDI 0487	No
Arm® Architecture Reference Manual Supplement Armv9, for Armv9-A architecture profile	DDI 0608	No

Table 1-3: Other publications

Document Name	Document ID
Advanced Encryption Standard (FIPS 197, November 2001)	-
Secure Hash Standard (SHS) (FIPS 180-4, August 2015)	-
Secure Hash Standard (SHS) (FIPS 202, August 2015)	-

Note

Arm tests its PDFs only in Adobe Acrobat and Acrobat Reader. Arm cannot guarantee the quality of its documents when used with any other PDF reader.

Adobe PDF reader products can be downloaded at http://www.adobe.com

2. Cryptographic extension support in the Cortex®-X3 core

The Cortex®-X3 core supports the optional Armv8.0-A and Arm®v8.2-A Cryptographic Extension.

The Armv8.0-A Cryptographic Extension adds A64 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.

The Arm®v8.2-A extensions, Armv8.2-A-SHA and Armv8.2-SM, add A64 instructions to accelerate SHA2-512, SHA3, SM3, and SM4.

The SVE2-AES, SVE2-SHA3, and SVE2-SM extensions add A64 instructions to accelerate SHA3, SM3, SM4, and AES encryption and decryption.

2.1 Product Revisions

The following table indicates the main differences in functionality between product revisions.

Table 2-1: Product revisions

Revision	Notes
r0p0	First release
r1p0	First release
r1p1	First release

Changes in functionality that have an impact on the documentation also appear in A.1 Revisions on page 15.

2.2 Disabling the Cryptographic Extension

Disabling of the Cryptographic Extension applies to all Cortex®-X3 cores in a cluster.

To disable the Cryptographic Extension, assert **CRYPTODISABLE**.

When **CRYPTODISABLE** is asserted:

- Executing a cryptographic instruction results in an **UNDEFINED** exception.
- ID_AA64ISAR0_EL1 indicates that the Cryptographic Extension is not implemented.

Related information

2.4 ID_AA64ISAR0_EL1, AArch64 Instruction Set Attribute Register 0 on page 11

2.3 Cryptographic Extensions register summary

Software can identify the cryptographic instructions that are implemented in the Cortex®-X3 core by reading the ID_AA64ISAR0_EL1 identification register.

The following table shows the instruction identification register for the Cortex®-X3 core Cryptographic Extension.

Table 2-2: Cryptographic Extension register summary

Name	Execution state	Description
ID_AA64ISAR0_EL1	AArch64	See 2.4 ID_AA64ISAR0_EL1, AArch64 Instruction Set Attribute Register 0 on page 11

2.4 ID_AA64ISAR0_EL1, AArch64 Instruction Set Attribute Register 0

Provides information about the instructions implemented in AArch64 state.

For general information about the interpretation of the ID registers, see 'Principles of the ID scheme for fields in ID registers'.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

Identification

Reset value

See individual bit resets.

Bit descriptions

Figure 2-1: AArch64_id_aa64isar0_el1 bit assignments

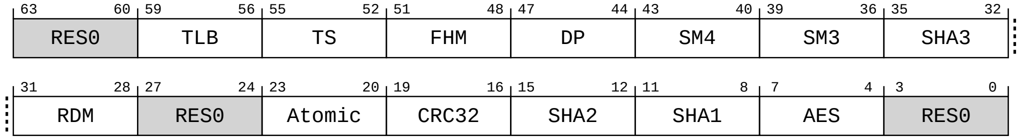


Table 2-3: ID_AA64ISAR0_EL1 bit descriptions

Bits	Name	Description	Reset
[63:60]	RES0	Reserved	0b0000
[59:56]	TLB	Indicates support for Outer Shareable and TLB range maintenance instructions. Defined values are: **0b0010** Outer Shareable and TLB range maintenance instructions are implemented.	
[55:52]	TS	Indicates support for flag manipulation instructions. Defined values are: **0b0010** CFINV, RMIF, SETF16, SETF8, AXFLAG, and XAFLAG instructions are implemented.	
[51:48]	FHM	Indicates support for FMLAL and FMLSL instructions. Defined values are: **0b0001** FMLAL and FMLSL instructions are implemented.	
[47:44]	DP	Indicates support for Dot Product instructions in AArch64 state. Defined values are: **0b0001** UDOT and SDOT instructions implemented.	
[43:40]	SM4	Indicates support for SM4 instructions in AArch64 state. Defined values are: **0b0000** When Cryptographic Extensions are not implemented or disabled then SM4 instructions are not implemented. **0b0001** When Cryptographic Extensions are implemented and enabled then SM4 instructions SM4E and SM4EKEY are implemented.	
[39:36]	SM3	Indicates support for SM3 instructions in AArch64 state. Defined values are: **0b0000** When Cryptographic Extensions are not implemented or disabled then SM3 instructions are not implemented. **0b0001** When Cryptographic Extensions are implemented and enabled then SM3 instructions SM3SS1, SM3TT1A, SM3TT1B, SM3TT2A, SM3TT2B, SM3PARTW1, and SM3PARTW2 are implemented.	
[35:32]	SHA3	Indicates support for SHA3 instructions in AArch64 state. Defined values are: **0b0000** When Cryptographic Extensions are not implemented or disabled then SHA3 instructions are not implemented. **0b0001** When Cryptographic Extensions are implemented and enabled then SHA3 instructions EOR3, RAX1, XAR, and BCAX are implemented.	
[31:28]	RDM	Indicates support for SQRDMLAH and SQRDMLSH instructions in AArch64 state. Defined values are: **0b0001** SQRDMLAH and SQRDMLSH instructions implemented.	
[27:24]	RES0	Reserved	0b0000
[23:20]	Atomic	Indicates support for Atomic instructions in AArch64 state. Defined values are: **0b0010** LDADD, LDCLR, LDEOR, LDSET, LDSMAX, LDSMIN, LDUMAX, LDUMIN, CAS, CASP, and SWP instructions implemented.	

Bits	Name	Description	Reset
[19:16]	CRC32	CRC32 instructions implemented in AArch64 state. Defined values are: **0b0001** CRC32B, CRC32H, CRC32W, CRC32X, CRC32CB, CRC32CH, CRC32CW, and CRC32CX instructions implemented.	
[15:12]	SHA2	SHA2 instructions implemented in AArch64 state. Defined values are: **0b0000** When Cryptographic Extensions are not implemented or disabled then SHA2 instructions are not implemented. **0b0010** When Cryptographic Extensions are implemented and enabled then SHA256H, SHA256H2, SHA256SU0, SHA256SU1, SHA512H, SHA512H2, SHA512SU0, and SHA512SU1 instructions are implemented. When the CRYPTO configuration parameter is true and the CRYPTODISABLE input is low at reset Cryptographic Extensions are implemented	
[11:8]	SHA1	SHA1 instructions implemented in AArch64 state. Defined values are: **0b0000** When Cryptographic Extensions are not implemented or disabled then SHA1 instructions are not implemented. **0b0001** When Cryptographic Extensions are implemented and enabled then SHA1C, SHA1P, SHA1M, SHA1H, SHA1SU0, and SHA1SU1 instructions are implemented. When the CRYPTO configuration parameter is true and the CRYPTODISABLE input is low at reset Cryptographic Extensions are implemented	
[7:4]	AES	AES instructions implemented in AArch64 state. Defined values are: **0b0000** SVE2-AES instructions are not implemented. This value is reported when Cryptographic Extensions are not implemented or are disabled. **0b0010** SVE2 AESE, AESD, AESMC, and AESIMC instructions are implemented plus SVE2 PMULLB and PMULLT instructions with 64-bit source. This value is reported when Cryptographic Extensions are implemented and enabled. When the CRYPTO configuration parameter is true and the CRYPTODISABLE input is low at reset Cryptographic Extensions are implemented	
[3:0]	RES0	Reserved	0b0000

Access

MRS <Xt>, ID_AA64ISAR0_EL1

<systemreg>	op0	op1	CRn	CRm	op2
ID_AA64ISAR0_EL1	0b11	0b000	0b0000	0b0110	0b000

Accessibility

MRS <Xt>, ID_AA64ISAR0_EL1

```
if PSTATE.EL == EL0 then
    if EL2Enabled() && HCR_EL2.TGE == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        AArch64.SystemAccessTrap(EL1, 0x18);
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TID3 == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return ID_AA64ISAR0_EL1;
elsif PSTATE.EL == EL2 then
    return ID_AA64ISAR0_EL1;
elsif PSTATE.EL == EL3 then
    return ID_AA64ISAR0_EL1;
```

Appendix A Document revisions

This appendix records the changes between released issues of this document.

A.1 Revisions

The first table is for the first release.

Then, each table compares the new issue of the book with the last released issue of the book. Release numbers match the revision history in Release Information on page 2.

Table A-1: Issue 0000-01

Change	Location
First beta release for r0p0	-

Table A-2: Differences between issue 0000-01 and issue 0000-02

Change	Location
First LAC release for r0p0	Revision history - no technical changes

Table A-3: Differences between issue 0000-02 and issue 0100-03

Change	Location
First EAC release for r1p0	Revision history
Minor editorial changes to ID_AA64ISAR0_EL1, AArch64 Instruction Set Attribute Register 0	2.4 ID_AA64ISAR0_EL1, AArch64 Instruction Set Attribute Register 0 on page 11

Table A-4: Differences between issue 0100-03 and issue 0101-04

Change	Location
First release for r1p1	Revision history - no technical changes