

# ARMv8-M processor power management

Version 1.0

**secure state protection**



# ARMv8-M processor power management

## secure state protection

Copyright © 2016 ARM Limited or its affiliates. All rights reserved.

### Release Information

### Document History

Issue	Date	Confidentiality	Change
0100	23 August 2016	Confidential	First release.

### Non-Confidential 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, and 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 REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF ANY USE OF THIS DOCUMENT, EVEN IF ARM HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

This document consists solely of commercial items. You shall be responsible for ensuring that any use, duplication or disclosure of this document complies fully with any relevant export laws and regulations to assure that this document or any portion thereof is not exported, directly or indirectly, in violation of such export laws. Use of the word “partner” in reference to ARM’s customers is not intended to create or refer to any partnership relationship with any other company. ARM may make changes to this document at any time and without notice.

If any of the provisions contained in these terms conflict with any of the provisions of any signed written agreement covering this document with ARM, then the signed written agreement prevails over and supersedes the conflicting provisions of these terms. This document may be translated into other languages for convenience, and you agree that if there is any conflict between the English version of this document and any translation, the terms of the English version of the Agreement shall prevail.

Words and logos marked with ® or ™ are registered trademarks or trademarks of ARM Limited or its affiliates in the EU and/or elsewhere. All rights reserved. Other brands and names mentioned in this document may be the trademarks of their respective owners. Please follow ARM’s trademark usage guidelines at <http://www.arm.com/about/trademark-usage-guidelines.php>

Copyright © 2016, ARM Limited or its affiliates. All rights reserved.

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

## ARMv8-M processor power management secure state protection

	<b>Preface</b>	
	<i>About this book</i> .....	6
	<i>Feedback</i> .....	8
<b>Chapter 1</b>	<b>Power management</b>	
	1.1 <i>Power management overview</i> .....	1-10
	1.2 <i>Protection measures</i> .....	1-11
	1.3 <i>Sleep mode</i> .....	1-12
	1.4 <i>Wakeup from sleep mode</i> .....	1-13

# Preface

This preface introduces the *ARMv8-M processor power management secure state protection*.

It contains the following:

- [About this book on page 6.](#)
- [Feedback on page 8.](#)

## About this book

Write a short description in the book map to render in the "About this book" section of the preface.

### Product revision status

The *rm**pn* identifier indicates the revision status of the product described in this book, for example, r1p2, where:

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

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

### Using this book

This book is organized into the following chapters:

#### Chapter 1 Power management

This section describes the security recommendations and events controlled by the SLEEPDEEP bit of the *System Control Register* (SCR).

### 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.

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>
```

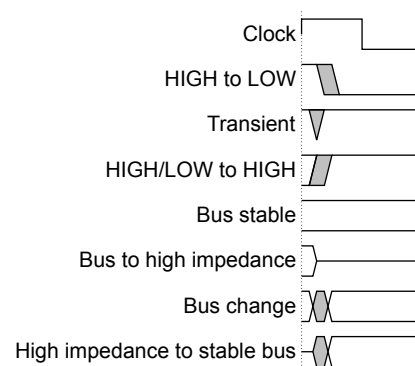
SMALL CAPITALS

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.

### 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.

### Lowercase n

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

## 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](mailto:errata@arm.com). Give:

- The title *ARMv8-M processor power management secure state protection*.
- The number ARM 100737\_0100\_0100\_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

## Power management

This section describes the security recommendations and events controlled by the SLEEPDEEP bit of the *System Control Register (SCR)*.

It contains the following sections:

- [1.1 Power management overview on page 1-10.](#)
- [1.2 Protection measures on page 1-11.](#)
- [1.3 Sleep mode on page 1-12.](#)
- [1.4 Wakeup from sleep mode on page 1-13.](#)

## 1.1 Power management overview

Clocks, resets, and power control hardware are critical aspects of a system that the Secure code relies on for correct operation.

The level and type of protection these resources require depends on the attack profile being defended against, for example, remote software attack or basic hardware attack. As a minimum, any control registers that can influence clocks, resets, or power domains that can affect software running in the Secure state, or peripherals that are used by Secure software, must only be accessible from the Secure state.

To support use cases where the Secure state is disabled, the protection of these registers is configurable.

## 1.2 Protection measures

System architects are encouraged to consider power-on reset, reset filtering, and different clock sources as extra protection measures.

### Power-on reset

Ensure that the power-on reset circuit can respond quickly enough to detect even short reductions in the power supply voltage. The voltage threshold must also be set so that a reset is always generated before the voltage drops low enough to cause data corruption in registers, combinational logic, or SRAM. As attackers often use unusual conditions, for example, low temperatures, you must make sure that the power-on reset either functions correctly at conditions outside the normal process corners, or exhibits fail-safe behavior, for example asserting reset.

### Reset filtering

One possible way in which an attacker might cause data corruption is to introduce a short glitch on an external reset signal. If the reset signal is not filtered correctly before being used internally, it is possible that the glitch only partially resets the device, which could result in security vulnerabilities. For immunity against cross talk and other types of noise, most devices already contain reset filters. However, in a similar way to the power-on reset, you must check the behavior of these circuits outside the normal operating conditions.

### Clocks

Invalid behavior of external clock sources can also be a source of data corruption that can lead to security vulnerabilities. The types of invalid behavior might include:

- Excessive jitter.
- Extreme mark-space ratios.
- Glitches.
- Clock frequencies outside the supported range.

In addition to over or under clocking the main clock, attackers might try to exploit the system by using invalid ratios between two clocks, for example, the main clock source, and a synchronous peripheral interface, which could result in a buffer over or under flow condition.

You can use several methods to defend against these attacks:

- Use trusted internal clock sources for critical operations.
- Clock conditioning circuitry that prevents critical digital logic being exposed to invalid clock behavior.
- Fail-safe behavior in the event of invalid clock signals, for example, asserting reset.

## 1.3 Sleep mode

The system can generate false wakeup events, for example, a debug operation can wake up the processor. Therefore, the software must be able to put the processor back into sleep mode after such an event. A program might have an idle loop to put the processor back into sleep mode.

One way to reduce energy use in the processor is to remove power, which removes both dynamic and static currents (sometimes called *power-gating*), or to stop the clock of the core which removes dynamic power consumption only and can be referred to as *clock-gating*.

The processor can have additional low-power states. These power states refer to the ability for the hardware *Phase Locked Loop* (PLL) and voltage regulators to be controlled by power management software. Deep sleep mode stops the system clock and switches off the PLL and flash memory.

The SLEEPDEEP bit of the *System Control Register* (SCR) selects which sleep mode is used.

This section contains the following subsections:

- [1.3.1 Wait for interrupt instruction \(WFI\) on page 1-12.](#)
- [1.3.2 Wait for event instruction \(WFE\) on page 1-12.](#)
- [1.3.3 Sleep-on-exit bit on page 1-12.](#)

### 1.3.1 Wait for interrupt instruction (WFI)

Wait For Interrupt is a hint instruction. It suspends execution, in the lowest power state available consistent with a fast wakeup without the need for software restoration, until a reset, asynchronous exception or other event occurs.

### 1.3.2 Wait for event instruction (WFE)

Wait For Event is a hint instruction. If the Event Register is clear, it suspends execution in the lowest power state available consistent with a fast wakeup without the need for software restoration, until a reset, exception or other event occurs.

When the processor executes a WFE instruction, it checks the value of the event register:

- 0** The processor stops executing instructions and enters sleep mode.
- 1** The processor clears the register to 0 and continues executing instructions without entering into sleep mode.

If the event register is 1, it indicates that the processor must not enter sleep mode on execution of a WFE instruction. Typically, two events can cause this behaviour:

- The external event signal is asserted.
- Another processor in the system has executed an SEV instruction.

### 1.3.3 Sleep-on-exit bit

If the SLEEPONEXIT bit of the SCR is set to 1 when the processor completes the execution of all queued exception handlers, it returns to Thread mode and immediately enters into sleep mode. Use this mechanism in applications that only require the processor to run when an exception occurs.

## 1.4 Wakeup from sleep mode

The conditions for the processor to wake up depend on the mechanism that causes it to enter into sleep mode.

This section contains the following subsections:

- [1.4.1 Wakeup from WFI or sleep-on-exit on page 1-13.](#)
- [1.4.2 Wakeup from WFE on page 1-13.](#)
- [1.4.3 Wakeup Interrupt Controller \(WIC\) on page 1-13.](#)
- [1.4.4 External event input signal on page 1-13.](#)
- [1.4.5 Power management programming hints on page 1-14.](#)

### 1.4.1 Wakeup from WFI or sleep-on-exit

Normally, the processor wakes up only when it detects an exception with sufficient priority to cause exception entry. Some embedded systems might have to execute system restore tasks after the processor wakes up, and before it executes an interrupt handler.

To achieve this behavior, set the PRIMASK bit to 1 and the FAULTMASK bit to 0. If an interrupt arrives that is enabled and has a higher priority than the current exception priority, the processor wakes up but does not execute the interrupt handler and continues sequential execution of the code until the processor sets PRIMASK to zero.

### 1.4.2 Wakeup from WFE

List of conditions which can cause the processor to wake up from WFE.

The processor wakes up if:

- It detects an exception with sufficient priority to cause exception entry, ignoring PRIMASK.
- It detects an external event signal.
- In a multiprocessor system, another processor in the system executes an SEV instruction.

In addition, if the SEVONPEND bit in the SCR is set to 1, any new pending interrupt triggers an event and wakes up the processor, even if the interrupt is disabled or has insufficient priority to cause exception entry.

### 1.4.3 Wakeup Interrupt Controller (WIC)

The *Wakeup Interrupt Controller* (WIC) is a peripheral that can detect an interrupt and wake the processor from deep sleep mode. The WIC is enabled only when the DEEPSLEEP bit in the SCR is set to 1.

The WIC is not programmable, and does not have any registers or user interface. It operates entirely from hardware signals.

When the WIC is enabled and the processor enters deep sleep mode, the power management unit in the system can power down most of the processor. This has the side effect of stopping the SysTick timer. When the WIC receives an interrupt, it takes several clock cycles to wake up the processor and restore its state, before it can process the interrupt. This behavior means that interrupt latency is increased in deep sleep mode.

————— **Note** —————

If the processor detects a connection to a debugger, it disables the WIC.

### 1.4.4 External event input signal

ARMv8-M processors provide an external event input signal. Peripherals can drive this signal, either to wake the processor from WFE, or to set the internal WFE event register to 1 to indicate that the processor must not enter sleep mode on a later WFE instruction.

### 1.4.5 Power management programming hints

The CMSIS provides two functions for WFE and WFI instructions.

```
void __WFE(void) // Wait for Event
```

```
void __WFI(void) // Wait for Interrupt
```