Cortex-M33 Cycle Model

Version 9.2

User Guide

ARM 101110_0902_00_en

Cortex-M33 Cycle Model

User Guide

Release Information

Document History

Issue	Date	Confidentiality	Change
0902-00	31 August 2017	Non-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*

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
Cortex-M33 Cycle Model User Guide

Preface

Chapter 1 Introduction

Chapter 2 Adding and configuring the SoC Designer component

Chapter 3 Available component ports

Chapter 4 Available component parameters

Chapter 5 Debug features

Preface

This preface introduces the *Cortex-M33 Cycle Model User Guide*.

It contains the following:

- *About this book* on page 7.
- *Feedback* on page 9.

About this book

This document describes how to use the Cortex-M33 Cycle Model in SoC Designer.

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

Using this book

This book is organized into the following chapters:

Chapter 1 Introduction

This section summarizes the functionality of the Cycle Model compared to that of the hardware, and describes the performance and accuracy of the Cycle Model. For details, see the *Cortex-M33 Technical Reference Manual* (Arm 100230).

Chapter 2 Adding and configuring the SoC Designer component

This section provides basic information about using the Cycle Model component. See the *SoC Designer User Guide* (Arm 100996) for more information.

Chapter 3 Available component ports

This section describes the differences between the pins in the hardware and those on the Cycle Model.

Chapter 4 Available component parameters

This section describes the component parameters and how to set them.

Chapter 5 Debug features

This section describes the debug features supported by the Cycle Model CADI debug interface.

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.

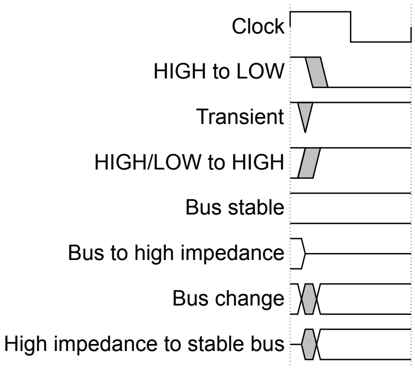

Figure 1 Key to timing diagram conventions

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.

Additional reading

ARM publications
Other publications

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 *Cortex-M33 Cycle Model User Guide*.
- The number ARM 101110_0902_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
Introduction

This section summarizes the functionality of the Cycle Model compared to that of the hardware, and describes the performance and accuracy of the Cycle Model. For details, see the *Cortex-M33 Technical Reference Manual* (Arm 100230).

It contains the following sections:

1.1 Supported hardware features

The following features of the Cortex-M33 hardware are fully implemented in the Cortex-M33 Cycle Model:

- Cortex-M33 Integer Core
- NVIC – Nested Vectored Interrupt Controller
- WIC – Wakeup Interrupt Controller Interface Support (support is for the interface only)
- Code AHB Interface (C-AHB)
- System AHB Interface (S-AHB)
- External PPB Interface (EPPB)
- Debug AHB Interface (D-AHB)
- BPU - Breakpoint Unit
- DWT – Debug Watchpoint and Trace
- MPU – Memory Protection Unit (optional)
- Floating Point Unit (optional) — single precision
- DSP Extension (optional)
- Security Extension (optional)
- SAU – Security Attribution Unit (optional)
- External coprocessor interface (optional)
- CTI - Cross Trigger Interface (optional)
- ITM and Data Watchpoint and Trace (optional)
- Hardware Profiling (ETM instruction execution trace, MTB instruction trace, Debugger integration)

1.2 Unsupported hardware features

The following features of the Cortex-M33 hardware are not implemented in the Cortex-M33 Cycle Model:

- SW/JTAG-DP
- Semihosting
- Restarting execution at a particular PC value
- DBGLVL is unsupported; it is set to the value 2 at build time.

1.3 Additional features for Cycle Model usability

The following features that are implemented in the Cycle Model do not exist in the hardware. These features have been added to the Cycle Model for enhanced usability.

- Run to Debug Point feature. This feature forces the debugger to advance the processor to the debug state instead of having the Cycle Model get into a nondebuggable state. See for more information.
- Waveform dumping using the waveform-related parameters described in Table 1-3 on page 21.
- Support for viewing registers (see *5.1 Register information* on page 5-30).
- Support for viewing memory (see *5.3 Memory view* on page 5-33).
- Support for disassembly view (see *5.2 Disassembly view* on page 5-32).
- Support for hardware profiling (see *5.4 Hardware profiling* on page 5-34).

Chapter 2
Adding and configuring the SoC Designer component

This section provides basic information about using the Cycle Model component. See the *SoC Designer User Guide* (Arm 100996) for more information.

It contains the following sections:

2.1 SoC Designer component files

The component files are the final output from the Cycle Model Studio compile and are the input to SoC Designer.

There are two versions of the component: an optimized release version for normal operation, and a debug version.

On Linux, the debug version of the component is compiled without optimizations and includes debug symbols for use with gdb. The release version is compiled without debug information and is optimized for performance.

On Windows, the debug version of the component is compiled referencing the debug runtime libraries so it can be linked with the debug version of SoC Designer. The release version is compiled referencing the release runtime library. Both release and debug versions generate debug symbols for use with the Visual C++ debugger on Windows.

The component files provided with your Cycle Model are:

Table 2-1 SoC Designer component files

Platform	File	Description
Linux	maxlib.lib<*component_name*>.conf	SoC Designer configuration file
	lib.<*component_name*>.mx.so	SoC Designer component runtime file
	lib<*component_name*>.mx_DBG.so	SoC Designer component debug file
Windows	maxlib<*component_name*>.lib.windows.conf	SoC Designer configuration file
	lib<*component_name*>.mx.dll	SoC Designer component runtime file
	lib<*component_name*>.mx_DBG.dll	SoC Designer component debug file

Additionally, this User Guide PDF is provided with the component.

2.2 Adding the Cycle Model to the component library

The compiled Cycle Model component is provided as a configuration file (`.conf`).

To make the component available in the Component Window in SoC Designer Canvas:

Procedure

1. Launch SoC Designer Canvas.
2. Select **File** > **Preferences**.
3. In the list on the left, click **Component Library**.
4. Under the **Additional Component Configuration Files** window, click **Add**.
5. Browse to the location where the Cycle Model is located and select the component configuration (.conf) file.
6. Click **OK**.
7. To save the preferences permanently, click the **OK & Save** button. The component is now available from the SoC Designer Component Window.
8. To pull the component onto the canvas, select and drag it.

The component's appearance may vary depending on your specific device configuration. Additional ports are provided depending on the model RTL configuration file (`default.conf`) used to create the Cycle Model.

Chapter 3
Available component ports

This section describes the differences between the pins in the hardware and those on the Cycle Model.

It contains the following sections:

3.1 Available ESL ports

Table 1-2 describes the ESL ports that are exposed in SoC Designer. See the *Cortex-M33 Technical Reference Manual* for more information.

Table 3-1 SoC Designer component files

ESL Port	Description	Type
AHB5Initiator_master_C_AHB	AHB Initiator Master Transactor	Transaction Master
AHB5Initiator_master_S_AHB	AHB Initiator Master Transactor	Transaction Master
APB4_master_EPPB	APB EPPB Master Transactor	Transaction Master
CLKIN	Free running clock	Clock Slave
clk-in	This port is used internally. Leave unconnected.	Clock Slave

3.2 Reset behavior and ports

The Cycle Model is reset internally each time SoC Designer Simulator is initialized. This behavior is standard and can not be changed. To view the internal reset sequence, set the **Align Waveforms** parameter to False, and this data appears in the waveform.

At simulation time zero and while simulation is running, you can generate a reset sequence. To do so, drive the reset pins on the component using external signals (for example, using the MxSigDriver component).

For information about reset pin names, bit ordering (for multiple cores), and required reset sequence, refer to the Technical Reference Manual for your IP.

3.3 Tied pins

The following pins are tied High:

- COREQREQn
- DBGEN
- DBGQREQn
- FPUQREQn
- HEXOKAYC
- MTBQREQn
- NSSTCLKEN
- SPIDEN
- SPNIDEN
- SSTCLKEN

Chapter 4
Available component parameters

This section describes the component parameters and how to set them.

It contains the following sections:

4.1 Changing parameter settings in SoC Designer

You can change the settings of all the component parameters in SoC Designer Canvas, and of some of the parameters in SoC Designer Simulator.

To modify the component parameters:

1. In SoC Designer Canvas, right-click on the component and select **Component Information**. You can also double-click the component. The **Edit Parameters** dialog box appears. The list of available parameters may differ slightly depending on the settings that you enabled in the configuration file (`default.conf`) when creating the component.
2. In the **Parameters** window, double-click the **Value** field of the parameter that you want to modify.
3. For text fields, type a new value. If a pulldown menu is available, select the desired value.

4.2 Component parameters

The following table describes the Cycle Model component parameters. Complete details about the parameters in this table are available in the Integration and Implementation Manual or Technical Reference Manual for your IP.

Table 4-1 Component parameters

Parameter name	Description	Allowed values	Default value	Init/ Runtime
AFVALIDE	ATB interface FIFO flush request (ETM instruction trace interface).	0, 1	0	Runtime
AFVALIDI	ATB interface FIFO flush request (ITM interface).	0, 1	0	Runtime
AHB5Initiator_master_C_AHB Align Data	AHB Lite Initiator Transactor Data Alignment.	true, false	false	Init
AHB5Initiator_master_C_AHB Big Endian	Endianness of data in AHB Transactor.	true, false	false	Init
AHB5Initiator_master_C_AHB Enable Debug Messages	Enables/disables port debug.	true, false	false	Runtime
AHB5Initiator_master_S_AHB Align Data	AHBP Transactor data alignment.	true, false	false	Init
AHB5Initiator_master_S_AHB Big Endian	Endianness of data in AHB Transactor.	true, false	false	Init
AHB5Initiator_master_S_AHB Enable Debug Messages	Enables/disables port debug.	true, false	false	Runtime
Align Waveforms	When set to true, waveforms dumped from the component are aligned with the SoC Designer simulation time. The reset sequence, however, is not included in the dumped data. When set to false, the reset sequence is dumped to the waveform data; however, the component time is not aligned with the SoC Designer time.	true, false	true	Init
APB4_master_EPPB Base Address	APB EPPB Master base address.	0x40000000 - 0x5FFFFFFF	0	Init
APB4_master_EPPB Enable Debug Messages	Enables/disables APB EPPB Master port debug.	true, false	false	Runtime
APB4_master_EPPB PReady Default High	EPPB Master Transactor PReady signal.	true, false	false	Runtime
APB4_master_EPPB Protocol Variant	Protocol Variant in use on APB EPPB Master port.	APB4	APB4	Init
APB4_master_EPPB Size	EPPB Transactor size region.	0 - 0x100000000	0	Init

Table 4-1 Component parameters (continued)

Parameter name	Description	Allowed values	Default value	Init/ Runtime
ATREADYE	ATDATA can be accepted (ETM instruction trace interface).	0, 1	0	Runtime
ATREADYI	ATDATA can be accepted (instruction trace).	0, 1	0	Runtime
ARM CycleModels DB Path	Sets the directory path to the database file.	Not used	empty	Init
CFGBIGEND	Static endianness setting. Refer to the *Arm Cortex-M33 Processor Integration and Implementation Manual* (Arm 100323) for more information.	0, 1	0	Runtime
CFGDSP	Enables support for Armv8-M DSP extensions.	0, 1	0	Runtime
CFGFPU	Enables support for hardware floating-point.	0, 1	0	Runtime
CFGNSSTCALIB	Specifies the Non-secure SysTick calibration.	Refer to the *Cortex-M33 Integration and Implementation Manual*.	0x207A11F	Init
CFGSECEXT	Enable support for Armv8-M security extension, if configured.	0, 1	0	Runtime
CFGSSTCALIB	Specifies the Secure SysTick calibration.	Refer to the *Cortex-M33 Integration and Implementation Manual*.	0x207A11F	Init
CORERET	Specifies that retention is supported in Core power domain.	0, 1	0	Runtime
CPERROR	Indicates that the coprocessor is not present or the instruction is not supported.	0, 1	0	Runtime
CPNSPRESENT	Indicates which Secure coprocessors are present in the system.	0, 1	0	Runtime
CPRDATA	Coprocessor read data bus.	0, 1	0	Runtime
CPREADY	Indicates whether the coprocessor is stalled or ready.	0, 1	0	Runtime
CPSPRESENT	Indicates which Secure coprocessors are present in the system.	0, 1	0	Runtime
CPUWAIT	Stalls the core out of reset.	0, 1	0	Runtime
CTICHIN	CTI Channel In.	0, 1	0	Runtime

Table 4-1 Component parameters (continued)

Parameter name	Description	Allowed values	Default value	Init/ Runtime
DBGRESTART	Request for synchronized exit from halt mode.	0, 1	0	Runtime
DBGRET	Debug power domain in retention.	0, 1	0	Runtime
DFTCGEN	Force all architectural clock gates open.	0, 1	0	Runtime
DFTRSTDISABLE	Synchronized multi-layer logic resets disabled.	0, 1	0	Runtime
Dump Waveforms	Determines whether SoC Designer dumps waveforms for this component. When enabled, SoC Designer writes accumulated waveforms to the waveform file in the following situations: when the waveform buffer fills, when validation is paused, when validation finishes, and at the end of each validation run.	true, false	false	Runtime
ECOREVNUM	Engineering change order revision numbering. Refer to the *Cortex-M33 Integration and Implementation Manual*.	0, 1	0	Runtime
EDBGRQ	External debug request. A debug agent in the system asserts this signal to request that the processor enters Debug state.	0, 1	0	Runtime
Enable Debug Messages	Determines whether debug messages are logged for the component.	true, false	false	Runtime
FPURET	FPU power domain in retention.	0, 1	0	Runtime
HADDRD	32-bit transfer address bus.	0, 1	0	Runtime
HADDRM	MTB Interface Transfer address.	0, 1	0	Runtime
HBURSTD	Indicates whether the transfer is part of a burst. This signal is ignored by the processor.	0, 1	0	Runtime
HBURSTM	MTB Interface Transfer burst length.	0, 1	0	Runtime
HEXOKAYS	Exclusive response. Data phase signal sampled on HREADYS that indicates whether the exclusive request was granted: 0 — Exclusive access has failed. 1 — Exclusive access is succesful.	0, 1	0	Runtime

Table 4-1 Component parameters (continued)

Parameter name	Description	Allowed values	Default value	Init/ Runtime
HNONSECD	Debug access security level request. See the *Cortex-M33 Integration and Implementation Manual* for details.	0, 1	0	Runtime
HNONSECM	MTB AHB security level request. When asserted, HNONSECM indicates a Nonsecure transfer.	0, 1	0	Runtime
HPROTD	Protection and outer memory attributes. Provides information on the access. HPROTD[0] is ignored by the processor; all debug transactions are treated as data accesses.	0, 1	0	Runtime
HPROTM	Protection and outer memory attributes.	0, 1	0	Runtime
HREADYM	Ready for MTB.	0, 1	0	Runtime
HSELM	Select access to MTB SRAM.	0, 1	0	Runtime
HSIZED	Indicates the size of the access. See the *Cortex-M33 Integration and Implementation Manual* for details.	0, 1	0	Runtime
HSIZEM	Transfer size.	0, 1	0	Runtime
HTRANSD	Indicates the type of current transfer. HTRANSD[0] is ignored by the processor; all transactions are treated as either Non-sequential or Idle.	0, 1	0	Runtime
HTRANSM	Transfer type.	0, 1	0	Runtime
HWDATAD	Data write bus (D-AHB interface).	0, 1	0	Runtime
HWDATAM	Write data (MTB interface).	0, 1	0	Runtime
HWRITED	Write transfer (D-AHB interface).	0, 1	0	Runtime
HWRITEM	Write transfer (MTB interface).	0, 1	0	Runtime
IDAUIDA, IDAUIDB	Region number. IDAUIDA[7:0] is the 8-bit region identifier associated with the IDAU region. The value is written to the IREGION field of the result register value, Rd[31:24], the destination register of a TT instruction when the instruction is executed in Secure state.	0, 1	0	Runtime

Table 4-1 Component parameters (continued)

Parameter name	Description	Allowed values	Default value	Init/ Runtime
IDAUIDVA, IDAUIDVB	Region number valid. IDAUIDVA indicates that the IDAU region number is valid. The value is written to the IRVALID field of the result register value, Rd[23], the destination register of a TT instruction when the instruction is executed in Secure state.	0, 1	0	Runtime
IDAUNCHKA, IDAUNCHKB	Region exempt from attribution check.	0, 1	0	Runtime
IDAUNSA, IDAUNSB	Non-secure region response.	0, 1	0	Runtime
IDAUNSCA, IDAUNSCB	Non-secure-callable region response.	0, 1	0	Runtime
IFLUSH	Flush instructions fetched on C-AHB or S-AHB on the previous HREADY cycle.	0, 1	0	Runtime
INITNSVTOR	Specifies the Non-secure vector table initialization value.	0, 1	0	Runtime
INITSVTOR	Specifies the Secure vector table initialization value.	0, 1	0	Runtime
IRQ	External interrupt signals.	Configuration-dependent	0	Runtime
LOCKNSMPU	Prevents changes to Nonsecure MPU memory regions already programmed.	0, 1	0	Runtime
LOCKNSVTOR	Prevents changes to the Non-secure vector table base address.	0, 1	0	Runtime
LOCKSAU	Prevents changes to Secure SAU memory regions already programmed.	0, 1	0	Runtime
LOCKSMPU	Prevents changes to programmed Secure MPU memory regions and all writes to the registers are ignored.	0, 1	0	Runtime
LOCKSVTAIRCR	Prevents changes to the Secure vector table base address, handling of Secure interrupt priority, BusFault, HardFault, and NMI security target settings in the processor.	0, 1	0	Runtime
MTBSRAMBASE	Location of MTB SRAM in processor memory map.	0, 1	0	Runtime
NIDEN	Configures event tracing.	0, 1	0	Runtime

Table 4-1 Component parameters (continued)

Parameter name	Description	Allowed values	Default value	Init/ Runtime
NMI	Non Maskable Interrupt.	0, 1	0	Runtime
NSSTCLKEN	Synchronous enable that is used with CLKIN to derive the Non-secure system SysTick clock.	0, 1	1	Runtime
RAMRD	RAM read data.	0, 1	0	Runtime
RXEV	Event in.	0, 1	0	Runtime
SAUDISABLE	Request to extend the processor sleeping state regardless of wake-up events. If the processor acknowledges this request driving SLEEPHOLDACKn LOW, this guarantees the processor remains idle even on receipt of a wake-up event.	0, 1	0	Runtime
SSTCLKEN	Synchronous enable that is used with CLKIN to derive the secure system SysTick clock.	0, 1	0	Runtime
SYNCREQE	Trace synchronization request from instruction trace sink.	0, 1	0	Runtime
SYNCREQI	Trace synchronization request from instruction trace sink.	0, 1	0	Runtime
TPIUACTV	TPIU data active.	0, 1	1	Runtime
TPIUBAUD	Unsynchronized TPIU baud indicator.	0, 1	0	Runtime
TSCLKCHANGE	Timestamp clock ratio change.	0, 1	1	Runtime
TSVALUEB	Global timestamp value.	0, 1	0	Runtime
Waveform File	Name of the waveform file.	`string`	arm_cm_CortexM33.vcd	Init
Waveform Format	The format of the waveform dump file.	VCD, FSDB	VCD	Init
Waveform Timescale	Sets the timescale to be used in the waveform.	Values in pulldown menu.	1 ns	Init
WICENREQ	Active HIGH request for deep sleep to be WIC-based deep sleep. Driven from the power management unit.	0, 1	0	Runtime

Chapter 5
Debug features

This section describes the debug features supported by the Cycle Model CADI debug interface.

It contains the following sections:

5.1 Register information

This section describes the supported register views.

Core registers

Table 5-1 Core registers

Name	Access
XPSR	Read/Write
R0-R15	R0 - R12 and R14 are Read/Write. R13 and R15 are Read Only.
R13_PROCESS_S	Read Only
R13_PROCESS_NS	Read/Write
R13_PROCESS	Read/Write
R13_MAIN_S	Read Only
R13_MAIN_NS	Read/Write
R13_MAIN	Read/Write
PRIMASK_S	Read Only
PRIMASK_NS	Read/Write
PRIMASK	Read/Write
InternalStatus	Read Only
FAULTMASK_S	Read Only
FAULTMASK_NS	Read/Write
FAULTMASK	Read/Write
CURRSTATE	Read Only
CONTROL_S	Read/Write
CONTROL_NS	Read/Write
CONTROL	Read/Write
BASEPRI_S	Read Only
BASEPRI_NS	Read Only
BASEPRI	Read/Write

System Control registers

Table 5-2 System Control registers

Name	Access
AIRCR_S	Read/Write
AIRCR_NS	Read/Write
AIRCR	Read/Write

DWT registers

Table 5-3 DWT registers

Name	Access
DWT_SLEEPCNT	Read/Write
DWT_LSUCNT	Read/Write
DWT_FOLDCNT	Read/Write
DWT_EXCCNT	Read/Write
DWT_CPICNT	Read/Write

5.2 Disassembly view

SoC Designer Simulator supports a disassembly view of a program running on the Cycle Model.

To display the disassembly view in SoC Designer Simulator, right-click on the Cycle Model and select **View Disassembly…** from the context menu. Refer to the *SoC Designer User Guide* (Arm 100996) for more information.

All CADI windows support breakpoints. When double-clicking on the proper location a red dot will indicate that a breakpoint is currently active. To remove the breakpoints simply doubleclick on the same location again.

5.3 Memory view

SoC Designer Simulator supports a memory view of a program running on the Cycle Model.

To display the memory view in SoC Designer Simulator, right-click on the Cycle Model and select **View Child Memory for…** from the context menu. Refer to the *SoC Designer User Guide* (Arm 100996) for more information.

5.4 Hardware profiling

Profiling data is enabled, and can be viewed using the Profiling Manager, which is accessible via the Debug menu in the SoC Designer Simulator.

The Cycle Model supports the following DWT profiling events. For more information, refer to the *Cortex-M33 Technical Reference Manual* (Arm 100234).

- DWT_SLEEPCNT
- DWT_LSUCNT
- DWT_FOLDCNT
- DWT_EXCCNT
- DWT_CPICNT