ARM® SystemC Cycle Models

User Guide

ARM DUI 1037A (ID111516)

ARM SystemC Cycle Models
User Guide

Release Information

The following changes have been made to this book.

Change history

Date	Issue	Confidentiality	Change
November 2016	A	Non-Confidential	First release.

Proprietary Notice

ARM Limited. Company 02557590 registered in England.

110 Fulbourn Road, Cambridge, England CB1 9NJ.

Product Status

The information in this document is final, that is for a developed product.

Web Address

http://www.arm.com

Contents
ARM SystemC Cycle Models User Guide

Preface

This preface introduces the *ARM® SystemC Cycle Models User Guide*. It contains the following sections:

- About this book
- Feedback

About this book

This book is for the ARM SystemC Cycle Models.

Intended audience

This book is written for experienced hardware engineers, software engineers and System-on-Chip (SoC) designers who might have experience of ARM products. You are expected to have experience of SystemC.

Using this book

This book is organized into the following chapters:

Chapter 1 ***Introduction***

Read this for information about SystemC Cycle Model supported platforms, compilers, and functionality.

Chapter 2 ***Using SystemC Cycle Models***

Read this for a description of how to use the SystemC Cycle Models with ARM CPAKs.

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.

The *ARM® Glossary* is available on the ARM Infocenter at, http://infocenter.arm.com/help/topic/com.arm.doc.aeg0014-/index.html.

Additional reading

This section lists publications by ARM and by third parties.

See Infocenter, http://infocenter.arm.com for access to ARM documentation.

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 DUI 1037A.
- 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 introduces the ARM SystemC Cycle Models. It contains the following sections:

1.1 About SystemC Cycle Models

ARM SystemC Cycle Models are compiled directly from the RTL code. The SystemC model wrappers are provided in source form to enable compiling for any SystemC 2.3.1-compliant simulator. You can integrate these models directly into any IEEE 1666-compliant SystemC environment.

Top-level recompilation of TLM wrappers is supported only within the CPAK environment. Pin-based SystemC Cycle Models may be used outside the scope of a CPAK.

1.1.1 Prerequisites

Simulation and recompilation require the Cycle Model Studio Runtime. Download the Cycle Model Studio Runtime Library installer from the Support area of ARM IP Exchange (https://www.armipexchange.com/). Linux and Windows versions of this runtime are available.

1.2 Supported platforms

SystemC Cycle Models are supported on Linux Red Hat EL 6.6 (64-bit) and above.

Only the Cortex-R52 SystemC model is supported on Windows 7 (64-bit).

1.3 TLM-based and pin-based support

All SystemC Cycle Models provide a SystemC signal level interface. Some models additionally provide SystemC wrappers which implement a SystemC TLM-2.0 interface for AXI and ACE. The TLM 2.0 wrappers allow for easier connection of the model into a system and interoperability with other TLM models.

No TLM interfaces are available for CHI, AHB, and APB.

Models included in an ARM SystemC CPAK must be either all TLM based or all pin-based. You can not mix TLM-based and pin-based models in the same CPAK system.

Note

Top-level recompilation of TLM wrappers is supported only within the CPAK environment. To recompile the model, place the SystemC Cycle Model you download from IP Exchange into the CPAK and follow the instructions in the README to recompile the Cycle Model. Pin-based SystemC Cycle Models may be used outside the scope of a CPAK.

1.4 Supported compilers

The SystemC Cycle Models have been tested with the following compiler versions:

- Linux — gcc 4.7.2. Newer versions of this compiler may also work. The SystemC Cycle Models include C++ Version 11 code, so the gcc in use must support this.
- Windows — Visual Studio 2013.

1.5 PMU support

PMU events are stored in C++ variables. For specifics about variable names for PMU events, refer to the files lib<*MODEL*>.system.hand lib<*MODEL*>.system.cpp and search for the ENABLE_PMU sections of code.

The following SystemC Cycle Models support PMU. Refer to the ARM *Techical Reference Manual* for your IP for details about profiled events.

- Cortex-A32
- Cortex-A35
- Cortex-A53
- Cortex-M7
- Cortex-M33
- Cortex-R8
- Cortex-R52

1.6 TARMAC Trace support

The TARMAC is controlled by environment variables and is slightly different for each CPU. Refer to the README file at the top level of a SystemC CPAK for more information.

The following SystemC Cycle Models support TARMAC Trace. Refer to *Configuring TARMAC Trace on page 2-27* for instructions on using TARMAC trace.

- Cortex-A32
- Cortex-A35
- Cortex-A53
- Cortex-M7
- Cortex-M23
- Cortex-M33
- Cortex-R8
- Cortex-R52

1.7 Package contents

Each SystemC Cycle Model contains the following files:

Note

All files may not be present for all models.

- <*MODEL*>Imp.cpp # This is the implementation for ARM TLM wrapper to connect the TLM ports to the pin level SystemC interface provided by lib<*MODEL*>.systemc.cpp/h.
- <*MODEL*>Imp.h # This is the header for ARM TLM wrapper to connect the TLM ports to the pin level SystemC interface provided by lib<*MODEL*>.systemc.cpp/h.
- <*MODEL*>ResetImp.cpp # This is the wrapper to connect the reset handler to the pin level SystemC interface provided by lib<*MODEL*>.systemc.cpp/h.
- <*MODEL*>.xmlAnswers # This file shows the configuration of the Cycle Model as requested on IP Exchange.

  Note

  For SystemC models without TLM ports, versions of the above files are provided in the NoTLM directory.

- lib<*MODEL*>.a # This is the RTL based core of the Cycle Model. This can be compiled into the system executable.
- lib<*MODEL*>.h # This is the base function header exposed by the core Cycle Model. This is required to access functions in the core Cycle Model. In the current case that is done by lib<MODEL>.systemc.cpp/h.
- lib<*MODEL*>.so # This is a compiled version of the ARM TLM wrapped Cycle Model implementation which can be linked into the system level model. This dumps PMU at the end of the simulation.
- lib<*MODEL*>_noPMU.so # This is a compiled version of the ARM TLM wrapped Cycle Model implementation which can be linked into the system level model.
- lib<*MODEL*>.systemc.cpp # This is the pin level systemC wrapping implementation around the core Cycle Model. This can be compiled to generate a signal level linked systemC model.
- lib<*MODEL*>.systemc.h # This is the pin level systemC wrapping header around the core Cycle Model. This can be compiled to generate a signal level linked systemC model.
- loadTCMUtil/* (Optional) # This is a Cortex-R8 specific set of files which are provided to load the TCM directly.
- Makefile # This is the makefile which can compile the ARM TLM level model into the shipped shared libraries.
- univent_tarmac.cpp # This is the univent interface implementation which can be hooked into the pin level Cycle Model to genreate univent traces.
- univent_tarmac.h # This is the univent interface header which can be hooked into the pin level Cycle Model to genreate univent traces.
- univentUtil/* # This is a folder containing model specific univent libraries which are needed to compile the univent_tarmac.cpp/h into the model.

Chapter 2
Using SystemC Cycle Models

This chapter includes the following sections:

Note

Top-level recompilation of TLM wrappers is supported only within the CPAK environment. Pin-based SystemC Cycle Models may be used outside the scope of a CPAK; regardless of type, it is strongly suggested that you download an ARM SystemC CPAK, which provides a valuable reference when you are working with SystemC Cycle Models. Access ARM System Exchange (https://www.armsystemexchange.com/cpaks/) to download a CPAK.

2.1 Replacing a SystemC Cycle Model in a CPAK

To replace IP in an existing ARM SystemC CPAK with an alternate configuration of the same IP — for example, swapping a single-core CPU for a dual-core CPU:

1. Access ARM IP Exchange (https://www.armipexchange.com/) to configure, build, and download the new SystemC Cycle Model. Ensure you select the **SystemC Model** tab as shown in Figure 2-1 (by default, the **SoC Designer** tab is active).

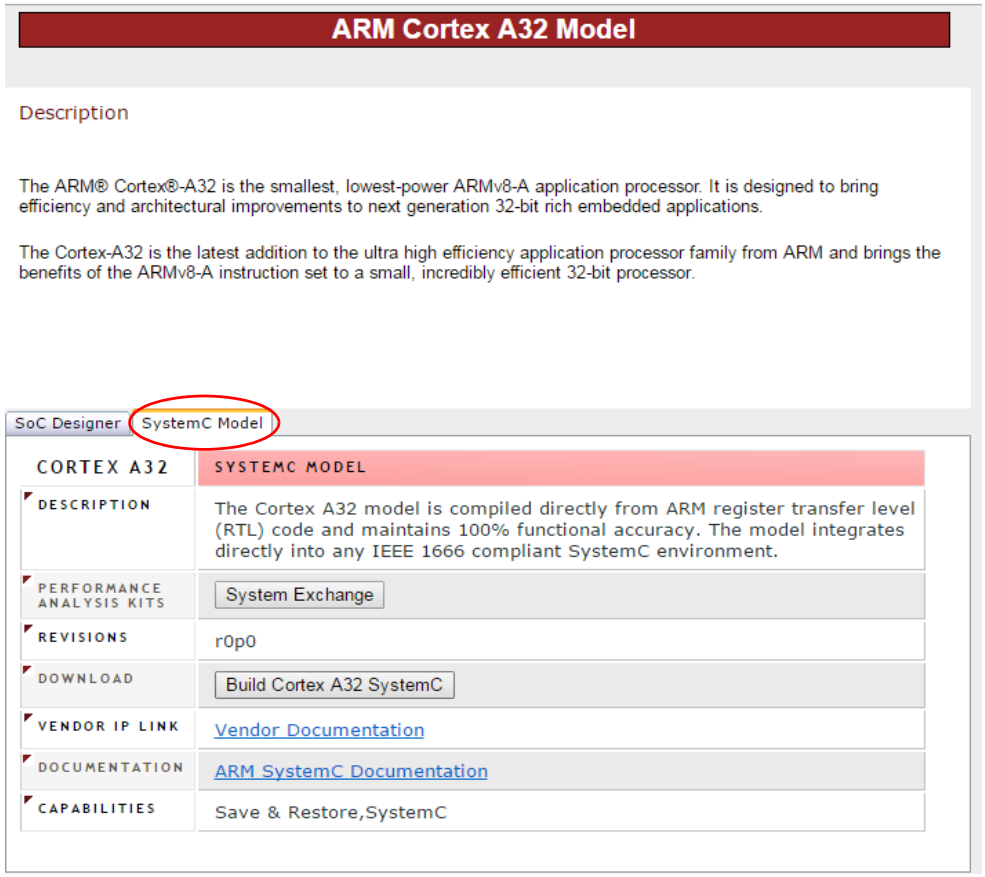

Figure 2-1 Building a SystemC Cycle Model on IP Exchange

2. In your CPAK directory, replace the contents of the MODELS directory with the files for your model.

2.2 Adding a SystemC Cycle Model to a CPAK

If you want to add new SystemC Cycle Models (for example, additional cores, memory, or peripheral components) to your SystemC CPAK:

1. Access ARM IP Exchange (https://www.armipexchange.com/) to configure, build and download the new model. Ensure you select the **SystemC Model** tab as shown in Figure 2-1 in the previous section (by default, the **SoC Designer** tab is active).
2. In your CPAK directory, add the files for the new SystemC Cycle Model to a new directory in CPAK/MODELS.

To connect two ports on different models:

1. Declare an sc_signal in the system_test.cpp file. This signal needs to be the same type and width as the two ports. If the ports are the same type but different widths, use scx_signal_sizer instead of sc_signal.
2. Edit the <*MODEL*>ResetImp.cpp file in the MODELS directory for both models and comment out the signal-to-port binding in the bind_nontlm_ports_signals method.
3. Bind the signal to both ports in the system_test.cpp file. For example:

```
sc_signal<bool> signal1;
 Inst1.port1.bind(signal1);
 Inst2.port1.bind(signal2);
```

4. Recompile the system. Models are recompiled automatically as part of the system recompile.

If you want to connect TLM ports, they must be the same protocol and width, otherwise a runtime error occurs. To connect two TLM ports on different models:

1. Bind the master port to the slave port. For example:

```
core.iSkt_M1->bind(bus.tSkt)
```

2. Recompile the system. Models are recompiled automatically as part of the system recompile.

2.3 Dumping Waveforms

You can instrument waveform dumping using the APIs documented in the *Cycle Model Studio SystemC User Manual* (ARM DUI 1057). If you do not have a full Cycle Model Studio installation that includes this document, it is available on ARM InfoCenter (http://infocenter.arm.com/help/index.jsp).

2.4 Configuring TARMAC Trace

By default, TARMAC Trace is enabled on SystemC Cycle Models. Disabling TARMAC Trace is done by means of environment variables and differs from IP to IP. The README.TXT file associated with the RTL Testbench for your IP includes instructions for disabling TARMAC Trace, as well as other information such as a description of the output format. ARM CPAKs include examples of TARMAC Trace usage (see the CPAK README.TXT file).

2.5 Resetting the SystemC Cycle Model

A default reset sequence is provided in source form in <*MODEL*>ResetImp.cpp, which you can modify as needed. Recompile the model after making your changes. Refer to the ARM Technical Reference Manual for your IP for details about its reset sequence.

Appendix A
Revisions

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

Table A-1 Issue A

Change	Location	Affects
First release for SystemC Cycle Models	-	-