APB System Controller Cycle Model

Version 9.0.0

User Guide

Non-Confidential

ARM®

ARM DUI 1052A (ID110716)

APB System Controller Cycle Model
User Guide

Release Information

The following changes have been made to this document.

Change History

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

Non-Confidential 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

Chapter 1.
Using the Cycle Model in SoC Designer

APB System Controller Cycle Model Overview 1-1
Cycle Model Features 1-2
Component Ports 1-3
Component Parameters 1-4
Debug Features 1-5
Register Information 1-5
Available Profiling Data 1-6

Preface

A Cycle Model component is a library that is generated through Cycle Model Studio™. The Cycle Model then can be used within a virtual platform tool, for example, SoC Designer.

About This Guide

This guide provides all the information needed to configure and use this Cycle Model in SoC Designer.

Audience

This guide is intended for experienced hardware and software developers who create components for use with SoC Designer. You should be familiar with the following products and technology:

- SoC Designer
- Hardware design verification
- Verilog or SystemVerilog programming language

Conventions

This guide uses the following conventions:

Convention	Description	Example
`courier`	Commands, functions, variables, routines, and code examples that are set apart from ordinary text.	`sparseMem_t SparseMemCreate-New();`
italic	New or unusual words or phrases appearing for the first time.	*Transactors* provide the entry and exit points for data ...
bold	Action that the user performs.	Click **Close** to close the dialog.
<text>	Values that you fill in, or that the system automatically supplies.	<platform>/ represents the name of various platforms.
[text]	Square brackets [ ] indicate optional text.	`$CARBON_HOME/bin/modelstudio [ <filename> ]`
[text1 | text2]	The vertical bar | indicates "OR," meaning that you can supply text1 or text 2.	`$CARBON_HOME/bin/modelstudio [<name>.symtab.db | <name>.ccfg ]`

Also note the following references:

- References to C code implicitly apply to C++ as well.
- File names ending in .cc, .cpp, or .cxx indicate a C++ source file.

Further reading

This section lists related publications.

The following publications provide information that relate directly to SoC Designer:

- *SoC Designer Installation Guide*
- *SoC Designer User Guide*
- *SoC Designer Standard Component Library Reference Manual*
- *SoC Designer AHBv2 Protocol Bundle User Guide*

The following publications provide reference information about ARM® products:

- *AMBA Design Kit Technical Reference Manual*
- *AMBA Specification*
- *AMBA AHB-Lite Protocol Specification*
- *AMBA AHB Protocol Specification*
- *Architecture Reference Manual*

See http://infocenter.arm.com/help/index.jsp for access to ARM documentation.

Glossary

Table 1:

AMBA	*Advanced Microcontroller Bus Architecture*. The ARM open standard on-chip bus specification that describes a strategy for the interconnection and management of functional blocks that make up a System-on-Chip (SoC).
AHB	*Advanced High-performance Bus*. A bus protocol with a fixed pipeline between address/control and data phases. It only supports a subset of the functionality provided by the AMBA AXI protocol.
APB	*Advanced Peripheral Bus*. A simpler bus protocol than AXI and AHB. It is designed for use with ancillary or general-purpose peripherals such as timers, interrupt controllers, UARTs, and I/O ports.
AXI	*Advanced eXtensible Interface*. A bus protocol that is targeted at high performance, high clock frequency system designs and includes a number of features that make it very suitable for high speed sub-micron interconnect.
Cycle Model	A software object created by the Cycle Model Studio from an RTL design. The Cycle Model contains a cycle- and register-accurate model of the hardware design.
Cycle Model Studio	Graphical tool for generating, validating, and executing hardware-accurate software models. It creates a *Cycle Model*, and it also takes a Cycle Model as input and generates a component that can be used in SoC Designer, Platform Architect, or Accellera SystemC for simulation.
CASI	*Cycle Accurate Simulation Interface*, is based on the SystemC communication library and manages the interconnection of components and communication between components.
CADI	*Cycle Accurate Debug Interface*, enables reading and writing memory and register values and also provides the interface to external debuggers.
CAPI	*Cycle Accurate Profiling Interface*, enables collecting historical data from a component and displaying the results in various formats.
Component	Building blocks used to create simulated systems. Components are connected together with unidirectional transaction-level or signal-level connections.
ESL	*Electronic System Level*. A type of design and verification methodology that models the behavior of an entire system using a high-level language such as C or C++.
HDL	*Hardware Description Language*. A language for formal description of electronic circuits, for example, Verilog.
RTL	*Register Transfer Level*. A high-level hardware description language (HDL) for defining digital circuits.

Table 1:

SoC Designer	The full name is *SoC Designer*. A high-performance, cycle accurate simulation framework which is targeted at System-on-a-Chip hardware and software debug, as well as architectural exploration.
SystemC	SystemC is a single, unified design and verification language that enables verification at the system level, independent of any detailed hardware and software implementation, as well as enabling co-verification with RTL design.
Transactor	*Transaction adaptors*. You add transactors to your component to connect your component directly to transaction level interface ports for your particular platform.

Chapter 1

Using the Cycle Model in SoC Designer

This chapter describes the functionality of the Cycle Model, and how to use it in SoC Designer Plus. It contains the following sections:

- APB System Controller Cycle Model Overview
- Component Ports
- Component Parameters
- Debug Features
- Available Profiling Data

1.1 APB System Controller Cycle Model Overview

The APB System Controller Cycle Model is a an APB peripheral component that provides control of the system boot behavior and low-power wait for interrupt mode. Basically, it is used to generate two signals, the Pause and the Remap signals.

The APB System Controller is provided as a single component, as described in Table 1-1:

Table 1-1 APB System Controller Component

Component	Description
CM_Sysctrl_Apb	Adds system control functionality to the system

1.1.1 Cycle Model Features

The APB System Controller provides the following features:

- Compliance to the AMBA (Rev 2.0) Specification.
- Register map control
- Wait for interrupt control

Note: The APB System Controller Cycle Model provides the functionality of the remap and pause controller (RemapPause) described in the AMBA Design Kit Technical Reference Manual.

1.2 Component Ports

The APB slave port interface is a multi-cycle transaction-based interfaces (CASI) that implements the APB protocol. Table 1-2 describes the ESL ports that are exposed in SoC Designer for the APB System Controller.

Table 1-2 ESL Component Ports

ESL Port	Description	Direction	Type
PRESETn	Reset port for resetting the System Controller component. If left unconnected it will reset. If it is connected it will use the connected value.	Input	Signal slave
apb	APB transaction slave port. The APB interface enables you to program the memory mapped registers.	Input	Transaction slave
nFIQ	FIQ interrupt input from the interrupt controller.	Input	Signal slave
nIRQ	IRQ interrupt input from the interrupt controller.	Input	Signal slave
clk-in	Input clock. The component is clocked at the frequency of the clock connected to the *clk-in* port. If the *clk-in* port is not connected, clocking is taken from SoC Designer System Properties.	Input	Clock slave
Pause	HIGH when in the wait for interrupt pause mode, and LOW at all other times.	Output	Signal master
Remap	HIGH when the normal memory map is in use, and LOW when the reset memory map is in use.	Output	Signal master

1.3 Component Parameters

You can change the settings of all the component parameters in SoC Designer Canvas, and of some of the parameters in SoC Designer Simulator. Table 1-3 describes the component parameters that are available for the APB System Controller.

Parameters that may be modified at runtime are identified with *Yes* in the *Runtime* column, otherwise the parameter values are fixed and must be set before the start of simulation.

Table 1-3 Component Parameters

Name	Description	Allowed Values	Default Value	Runtime
Align Waveforms	When set to *true*, waveforms dumped from components are aligned with SoC Designer simulation time. The reset sequence, however, is not included in dumped data. When set to *false*, the reset sequence is dumped to waveform data, however, the component time is not aligned with SoC Designer time.	true, false	true	No
apb Base Address	Base address for the APB region accessed via the APB slave port of the System Controller.	0x0 - 0xFFFFFFFF	0x0	No
apb Enable Debug Messages	When set to *true*, writes APB debug messages to the SoC Designer output window.	true, false	false	Yes
apb Size	Address range size.	0x0 - 0xFFFFFFFF	0x100000000	No
Carbon DB Path	Sets the directory path to the database file.	Not Used	empty	No
Dump Waveforms	Whether SoC Designer dumps waveforms for this component.	true, false	false	Yes
Enable Debug Messages	Enable or disable the capture of debug messages for the component.	true, false	false	Yes
Waveform File [1]	Name of the waveform file.	*string*	arm_cm_CM_Sysctrl_Apb.vcd	No
Waveform Format	The format of the waveform dump file.	VCD, FSDB	VCD	No
Waveform Timescale	Sets the timescale to be used in the waveform.	Many values in drop-down	1 ns	No

1. When enabled, SoC Designer writes accumulated waveforms to the waveform file in the following situations: when the waveform buffer fills, when validation is paused and when validation finishes, and at the end of each validation run.

1.4 Debug Features

The APB System Controller has a debug interface (CADI) that allows you to view, manipulate, and control the registers in the SoC Designer Simulator and Model Debugger. A view can be accessed in SoC Designer by right-clicking on the component and choosing the appropriate menu entry.

Transactions can be visualized using the transaction monitors attached to connections. By right clicking on any of the APB connections in SoC Designer, a transaction monitor probe can be attached.

1.4.1 Register Information

The APB System Controller Cycle Model has four sets of registers that are accessible via the debug interface. Registers are grouped into sets according to functional area.

- APB Port Signals
- General Registers
- Peripheral ID Registers
- PrimeCell ID Registers

Note that the *apb_sig* tab is a group of signals, not registers.

1.4.1.1 APB Port Signals

Table 1-4 shows the APB port signals. See the *ARM AMBA 3 APB Protocol Specification* for more information about these signals.

Table 1-4 APB Port Signals

Name	Description	Type
PRDATA	Unidirectional AMBA APB Read Data bus. Performs read cycles when PWRITE is LOW. This bus can be up to 32-bits wide.	read-only
PADDR	Subset of the AMBA APB address bus. It can be up to 32 bits wide.	read-only
PWDATA	Unidirectional AMBA APB Write Data bus. Performs write cycles when PWRITE is HIGH. This bus can be up to 32 bits wide.	read-only
PWRITE	AMBA APB transfer direction signal, indicates a write access when HIGH, read access when LOW.	read-only
PSEL	System Controller module Select signal from the decoder within the APB bridge. When HIGH this signal indicates the slave device is selected by the APB bridge, and that a data transfer is required.	read-only
PENABLE	AMBA APB enable signal. PENABLE is asserted HIGH for one cycle of PCLK to enable a bus transfer.	read-only

1.4.1.2 General Registers

Table 1-5 shows the General registers.

Table 1-5 General Registers Summary

Register	Description	Type
Pause	Pause register. Any write sets the Pause register. A system reset will clear this register	write-only
Remap	Remap register. Any write sets the Remap register. A system reset will clear this register.	read-write
ResetStatus	System Reset Status register. Specify bits to be set in the reset status register (bits[7:1]).	read-write
ResetStatusClr	Reset Status Clear register. Specify bits to be cleared in the reset status register (bits[7:0]).	write-only

1.4.1.3 Peripheral ID Registers

Table 1-6 shows the Peripheral Identification registers.

Table 1-6 Peripheral ID Registers Summary

Register	Description	Type
RpcPeriphID0	Identifies the part number of the peripheral.	read-only
RpcPeriphID1	Identifies the part number and designer of the peripheral.	read-only
RpcPeriphID2	Identifies the revision and designer of the peripheral.	read-only
RpcPeriphID3	Identifies the configuration of the peripheral.	read-only

1.4.1.4 PrimeCell ID Registers

Table 1-7 shows the PrimeCell Identification registers.

Table 1-7 PrimeCell ID Registers Summary

Register	Description	Type
RpcPCellID0	Determines the reset value.	read-only
RpcPCellID1	Determines the reset value.	read-only
RpcPCellID2	Determines the reset value.	read-only
RpcPCellID3	Determines the reset value.	read-only

1.5 Available Profiling Data

This Cycle Model does not provide profiling streams, and therefore does not provide profiling information. Transaction related information can be retrieved from the respective bus components.