PrimeCell AHB Bus Matrix BP010 Cycle Model

Version 9.1.0

User Guide

ARM®

ARM DUI1082A (ID121516)

PrimeCell AHB Bus Matrix BP010 Cycle Model
User Guide

Release Information

The following changes have been made to this document.

Change History

Date	Issue	Confidentiality	Change
February 2017	A	Non-Confidential	Restamp 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

Preface

A Cycle Model component is a library developed from ARM intellectual property (IP) 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 the 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*

The following publications provide reference information about ARM® products:

- *AMBA 3 AHB-Lite Overview*
- *AMBA Specification (Rev 2.0)*
- *AMBA AHB Transaction Level Modeling Specification*
- *Architecture Reference Manual*

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

The following publications provide additional information on simulation:

- IEEE 1666™ SystemC Language Reference Manual, (IEEE Standards Association)
- SPIRIT User Guide, Revision 1.2, SPIRIT Consortium.

Glossary

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 (or *Cycle Model Compiler*) 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	*ESL API Simulation Interface*, is based on the SystemC communication library and manages the interconnection of components and communication between components.
CADI	*ESL API Debug Interface*, enables reading and writing memory and register values and also provides the interface to external debuggers.
CAPI	*ESL API 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.
SoC Designer	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 component, and how to use it in SoC Designer. It contains the following sections:

1.1 BP010 Cycle Model Functionality

The Bus Matrix Cycle Model is a configurable component that enables multiple AHB masters to be connected to multiple AHB slaves. Once configured through the use of the AMBA Designer Graphical User Interface configuration tool the RTL design files are generated. These files define a specific configuration of the Bus Matrix, including a specification of the address map information. It is this configuration that is converted to a Cycle Model.

Use the AMBA Designer tool to design variants of your Bus Matrix. You can then generate, test, and profile complex AMBA bus systems in:

- a transaction-level modeling environment
- Verilog

This section provides a summary of the functionality of the Cycle Model compared to that of the hardware, and the performance and accuracy of the Cycle Model.

- Implemented Hardware Features
- Unsupported Hardware Features
- Features Additional to the Hardware

1.1.1 Implemented Hardware Features

The Bus Matrix is a highly configurable infrastructure component that supports:

- 1-16 AHB-Lite slave ports
- 1-16 AHB-Lite master ports
- Data widths of 32 or 64 bits
- Address widths of 32 or 64 bits
- Architecture type, AHB and ARM11 extensions:
  - AHB2, support and AHB2.0 interface
  - V6, support all ARM11 AHB extensions
  - Excl(usive), support the ARM11 exclusive access extensions only
  - Unalign, support the ARM11 unaligned access extensions only
- Arbiter types: round robin, fixed and burst
- Optional xUSER signals with widths between 0 and 32 bits (inclusive)
- Address map with REMAP support and a default destination for unmapped address ranges

1.1.2 Unsupported Hardware Features

The following features of the BP010 hardware are not implemented in the Cycle Model:

- None. (All hardware features are fully supported)

1.1.3 Features Additional to the Hardware

The following features that are implemented in the BP010 Cycle Model do not exist in the BP010 hardware. These features have been added to the Cycle Model for enhanced usability.

- The REMAP input can be controlled by a wired connection, or if it is left unconnected the value of a parameter is used.

1.2 Adding and Configuring the SoC Designer Component

The following topics briefly describe how to use the component. See the *SoC Designer User Guide* for more information.

- SoC Designer Component Files
- Adding the Cycle Model to the Component Library
- Adding the Component to the SoC Designer Canvas

1.2.1 SoC Designer Component Files

The component files are the final output from the Cycle Model Studio compile and are the input to SoC Designer Plus. 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 Plus. 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 provided component files are listed below:

Table 1-1 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.lib<*component_name*>.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 file is provided with the component.

1.2.2 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, perform the following steps:

1. Launch SoC Designer Canvas.
2. From the *File* menu, select **Preferences**.
3. Click on **Component Library** in the list on the left.
4. Under the *Additional Component Configuration Files* window, click **Add**.
5. Browse to the location where the SoC Designer Cycle Model is located and select the component configuration file:
   - `maxlib.lib<component_name>.conf` (for Linux)
   - `maxlib.lib<component_name>.windows.conf` (for Windows)
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.*

1.2.3 Adding the Component to the SoC Designer Canvas

Locate the component in the *Component Window* and drag it out to the Canvas.

This figure shows a configuration of the Bus Matrix with 2 slave and 2 master ports. The xUSER width is 0 so none of the USER signals appear. Your component may appear with fewer or more ports, depending on how it was configured in AMBA Designer.

Depending on how you configured the ports in AMBA Designer, the port names may be more descriptive as to the name of the device to which the port will be connected, and the protocol type. The names are fully customizable in AMBA Designer.

1.3 Available Component ESL Ports

Table 1-2 describes the ESL ports of the component, created by AMBA Designer, that are exposed in SoC Designer Plus.

Table 1-2 ESL Component Ports

ESL Port	Description	Direction	Type
HCLK	Clock signal for the AHB clock domain.	input	Clock slave
HRESETn	Reset signal.	input	Signal slave
REMAP	This is a 4-bit port selects the remap entry that is currently active	input	Signal slave
clk-in	Input clock. When using the HCLK port, the *clk-in* port should not be connected.	input	Clock slave
Slave Ports	There is a slave port created for each port defined in AMBA Designer. For example, *s00_ahb_32*.	slave	AHB_LITE
Master Ports	There is a master port created for each port defined in AMBA Designer. For example, *m00_ahb_32*. The type is really a “slave gasket”, see the TRM for more information.	master	AHB_LITE
HAUSER<*portname*> HWUSER<*portname*> HRUSER<*portname*>	If the BP010 was configured with a USER width greater than 0 then 3 USER ports are created for each defined Slave port and 3 USER ports are created for each defined Master port.	master/slave	Signal master/slave

All pins that are not listed in this table have been either tied or disconnected for performance reasons.

Note: Some ESL component port values can be set using a component parameter. This includes the RESETn and REMAP ports. In those cases, the parameter value will be used whenever the ESL port is not connected. If the port is connected, the connection value takes precedence over the parameter value.

1.4 Setting 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. To modify the Cycle Model parameters:

1. In the Canvas, right-click on the Cycle Model and select **Edit Parameters...**. You can also double-click the component. The *Edit Parameters* dialog box appears.
2. In the *Parameters* window, double-click the **Value** field of the parameter that you want to modify.
3. If it is a text field, type a new value in the *Value* field. If a menu choice is offered, select the desired option. The parameters are described in Table 1-3.

Table 1-3 Component Parameters

Name	Description	Allowed Values	Default Value	Runtime[1]
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	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	Whether debug messages are logged for the component.	true, false	false	Yes
RESETn	Sets the value for the Reset signal.	0x0, 0x1	0x1	Yes
REMAP	Sets the value for the REMAP signal.	0x0, 0x1,0x2, 0x4,0x8	0x0	Yes
<*master port name*>_ size[0-5] [2]	Sizes of memory regions.	0x0 - 0x100000000	size0 default is 0x100000000, size1-5 default is 0	No
<*master port name*>_ start[0-5] [2]	Start addresses of memory regions.	0x0 - 0xffffffff	0x00000000	No
<*master port name*> Enable Debug Messages	Whether debug messages are logged for the master ports. There is one parameter for each master port.	true, false	false	Yes

Table 1-3 Component Parameters (continued)

Name	Description	Allowed Values	Default Value	Runtime[1]
<*slave port name*> Enable Debug Messages	Whether debug messages are logged for the slave ports. There is one parameter for each slave port.	true, false	false	Yes
Waveform File [3]	Name of the waveform file.	*string*	arm_cm_<*component_name*>.vcd	No
Waveform Timescale	Sets the timescale to be used in the waveform.	Many values in drop-down	1 ns	No

1. *Yes* means the parameter can be dynamically changed during simulation, *No* means it can be changed only when building the system, *Reset* means it can be changed during simulation, but its new value will be taken into account *only* at the next reset.
2. The square brackets used in parameter names specify a range of numbers that are available for the Cycle Model. The parameter name for the start addresses “s_[0-1]_start[0-5]” for example will be expanded to 12 possible parameter name combinations that range from “s_0_start0” to “s_1_start5”. The size of a memory region depends on the “s[N]_start[M]” and “s[N]_size[M]” parameters. The end address is calculated as StartAddr +Size -1. The size of the memory region must not exceed the value of 0x100000000. If the sum of StartAddr+Size is greater than 0x100000000, the size of the memory region is reduced to the difference: 0x100000000-StartAddr.
3. 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.

1.5 Debug Features

The BP010 Cycle Model has no internal registers or memories that are visible in the programmers view. However, to aid in debugging systems the AHBLite signals, and the xUSER signals for each AHB port are displayed on individual tabs in the register view. Access this view in SoC Designer by right clicking on the Cycle Model and choosing the appropriate menu entry.

1.5.1 Register Information

The slave and master port registers allow you to examine the values of the listed signals.

1.6 Available Profiling Data

The BP010 Cycle Model component has no profiling capabilities.