PrimeCell® Color LCD Controller (PL111) Cycle Model

Version 9.1.0

User Guide

Non-Confidential

ARM DUI 1065A (ID120216)

PrimeCell® Color LCD Controller (PL111) 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

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:

- PL111 Cycle Model Functionality
- Available Component ESL Ports
- Setting Component Parameters
- Debug Features
- Available Profiling Data

1.1 PL111 Cycle Model Functionality

The PrimeCell Color LCD Controller is an AMBA compliant master-slave module that connects to the Advanced High-performance Bus (AHB). The LCD Controller provides all the necessary control signals to interface directly to a variety of color and monochrome LCD panels.

This section provides a summary of the functionality of the Cycle Model compared to that of the hardware. For details of the functionality of the hardware that the Cycle Model simulates, refer to the *ARM PrimeCell Color LCD Controller (PL111) Technical Reference Manual*.

- Fully Functional Features
- Unsupported Hardware Features
- Features Additional to the Hardware

1.1.1 Fully Functional Features

The following features of the PL111 hardware implementation are fully implemented in the PL111 Cycle Model.

- it complies with the AMBA2 specification for the two AHB interfaces
- a 64 bit master AHB interfaces is used to access the frame buffer
- dual 16-deep programmable 64 bit FIFOs are used for buffering incoming display data
- single- or dual-panel mono Super Twisted Nematic (STN) displays with 4 or 8-bit interfaces
- single- and dual-panel color STN displays
- Thin Film Transistor (TFT) color panels
- programmable resolution up to 1024x768
- hardware cursor support for single-panel displays
- 15 gray-level mono, 3375 color STN, and 32K color palettized TFT support
- 1,2, or 4 bits-per-pixel (bpp) palettized displays for mono STN
- 1, 2, 4, or 8bpp palettized color displays for color STN and TFT
- 16bpp true-color non-palettized, for color STN and TFT
- 24bpp true color non-palettized, for color TFT
- programmable timing for different display panels
- 256 entry, 16-bit palette ram, arranged as a 128x32 bit RAM physically
- frame, line, and pixel clock signals
- AC bias signal for STN, data enable signal for TFT panels
- patented gray scale algorithm
- supports little and big-endian, and Windows® CE data formats

1.1.2 Unsupported Hardware Features

The following features of the PL111 hardware are not implemented in the PL111 Cycle Model:

- support for the scan pins
- support for the *Integration Test Control* Register

1.1.3 Features Additional to the Hardware

The following features have been added to the PL111 Cycle Model to enhance usability and do not exist in the PL111 hardware:

- The optional Integrated LCD Simulator. This feature only supports TFT type panels.
- A simulation model for the palette and cursor rams is automatically defined and included in the PL111 Cycle Model

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. 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 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 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.

1.3 Available Component ESL Ports

Table 1-2 describes the ESL ports that are exposed in SoC Designer. See the *ARM PrimeCell® Color LCD Controller (PL111) Technical Reference Manual* for more information..

Table 1-2 ESL Component Ports

ESL Port	Description	Direction	Type
ahb_slave ahb_master	AHB slave and master ports.	slave master	AHB
HRESETn	Input reset for AHB bus signals. Reset port for receiving reset signal.	Input	Signal slave
CLCDCLK	Clock for LCD Controller main blocks.	Input	Clock Slave
HCLK	Clock for AHB bus	Input	Clock Slave
nCLCCLKRESET	Reset signal for CLCDCLK domain	Input	Signal slave
clk-in	Input clock. This component can be connected to the clock master. If it is unconnected, it is implicitly connected to the system master clock.	Input	Clock slave
CLCDCLKSEL	Clock source select, Bit 5 of *LCDTiming register 2* drives this signal	Output	Signal master
CLCDMBEINTR	Bus error interrupt	Output	Signal master
CLCDFUFINTR	DMA FIFO underflow interrupt	Output	Signal master
CLCDLNBUINTR	Next base address update interrupt	Output	Signal master
CLCDVCOMPINTR	Vertical compare interrupt	Output	Signal master
CLCDINTR	Combined interrupt	Output	Signal master
The following signals are included if the integrated LCD simulator is not selected			
CLPOWER	LCD Panel Power enable	Output	Signal master
CLLP	Line sync Pulse (STN)/ Horiz Sync pulse (TFT)	Output	Signal master
CLCP	LCD panel clock	Output	Signal master
CLFP	Frame pulse (STN)/ Vertical Sync pulse (TFT)	Output	Signal master
CLAC	STN AC bias drive or TFT data enable output	Output	Signal master
CLD[23:0]	LCD panel data	Output	Signal master
CLLE	Line End signal	Output	Signal master

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

1.3.1 AHB Transaction Slave/Master Interfaces

The AHB_slave port and the AHB_master port are AHB transaction ports. The AHB_slave port is 32 bits wide and is used for configuring the PL111. The AHB_master port is 64 bits wide and normally connected to an AHB bus that is also connected to a memory component. This is the bus that is used for DMA transfers from the image buffer to the PL111.

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 component's parameters:

1. In the Canvas, right-click on the component 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
HRESETn	Specifies the value used for the HRESETn input if it is left disconnected.	0,1	1	Yes
nCLCLKRESET	Specifies the value used for the nCLCLKRESET input if it is left disconnected.	0,1	1	Yes
ahb_slave Enable Debug Messages ahb_master Enable Debug Messages	Whether debug messages are logged for the *ahb_slave* and *ahb_master* ports.	true, false	false	Yes
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.	true, false	false	Yes

Table 1-3 Component Parameters (continued)

Name	Description	Allowed Values	Default Value	Runtime[1]
LCD_resolution	Specify the maximum resolution that the Cycle Model supports.	*sizex*X*sizey*	1024X768	No
LCD_update_divider	Specify the number of times the PL111 indicates it has updated the display buffer before the integrated display is updated. Larger numbers here may speed up simulation time.	integer	1	Yes
Waveform File [2]	Name of the waveform file.	*string*	arm_cm_PL111.vcd	No
Waveform Format	Sets the format of the waveform 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. *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. 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

Setting the `Enable Debug Messages` parameter to true will enable additional checking of the PL111 configuration. Some incorrect configurations will be reported with this setting.

The PL111 contains a set of memory mapped configuration registers. See the *ARM PrimeCell® Color LCD Controller (PL111) Technical Reference Manual* for details on the specific addresses and their functions.

The PL111 Cycle Model has a debug interface (CADI) that allows the user to view, manipulate, and control the registers and memory. A view can be accessed in SoC Designer by right clicking on the Cycle Model and choosing the appropriate menu entry.

1.5.1 Register Information

The PL111 Cycle Model are the same as the memory mapped registers described in the *PL111 Technical Reference Manual*. They are divided into three groups:

- LCD Registers. These provide control over most functions of the PL111
- Cursor Registers. These provide control over the hardware cursor in the PL111.
- ID Registers. These show the PeriphID and CellID values. These are read-only registers.

1.5.2 Memory Information

The PL111 includes two internal memories that can be viewed by right clicking on the component and selecting `View Memory...`

The LCDPalette view shows the contents of the Palette memory. This memory is physically structured as a 128x32 memory, and it is displayed with the same structure.

The Cursor memory is physically structured as a 256x32 memory. There are two views of this memory provided, and both use the 256x32 layout. The view named Cursoriamge_raw_view shows the values as they are stored in the internal memory. The same memory contents is also displayed by the Cursorimage_LBBP_layout_view but in this case the data is formatted using the LBBP (little-endian byte, big-endian pixel). This latter view allows for simpler checking of the memory contents as it matches the format used in the diagrams of the *PL111 Technical Reference Manual.*

If the integrated LCD simulator option was selected then the PL111 Cycle Model also includes a third memory that contains the image data that is displayed in the LCD simulator window. The contents of this memory is updated and displayed at each vertical pulse generated by the PL111. The contents of this memory is the image that is sent to the LCD panel and it will include the cursor overlay, if the hardware cursor feature of the PL111 has been enabled and the cursor is within the visible region of the display.

1.6 Available Profiling Data

The PL111 Cycle Model component has no profiling capabilities.