Core Tile for ARM11™ MPCore™

HBI-0146

User Guide

ARM®

Core Tile for ARM11 MPCore
User Guide

Release Information

The following changes have been made to this book.

Change History

Date	Issue	Confidentiality	Change
October 2005	A	Non-Confidential	First release
December 2005	B	Non -Confidential	Fixes for Errata
September 2006	C	Non-Confidential	Fixes for Errata
September 2008	D	Non-Confidential	Fixes for Errata
January 2009	E	Non-Confidential	Fixes for Errata
July 2010	F	Non-Confidential	Document update

Proprietary Notice

Words and logos marked with® or ™ are registered trademarks or trademarks of ARM Limited in the EU and other countries, except as otherwise stated below in this proprietary notice. Other brands and names mentioned herein may be the trademarks of their respective owners.

Neither the whole nor any part of the information contained in, or the product described in, this document may be adapted or reproduced in any material form except with the prior written permission of the copyright holder.

The product described in this document is subject to continuous developments and improvements. All particulars of the product and its use contained in this document are given by ARM Limited in good faith. However, all warranties implied or expressed, including but not limited to implied warranties of merchantability, or fitness for purpose, are excluded.

This document is intended only to assist the reader in the use of the product. ARM Limited shall not be liable for any loss or damage arising from the use of any information in this document, or any error or omission in such information, or any incorrect use of the product.

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`

Conformance Notices

This section contains conformance notices.

Federal Communications Commission Notice

This device is test equipment and consequently is exempt from part 15 of the FCC Rules under section 15.103 (c).

CE Declaration of Conformity

The system should be powered down when not in use.

The Core Tile generates, uses, and can radiate radio frequency energy and may cause harmful interference to radio communications. However, there is no guarantee that interference will not occur in a particular installation. If this equipment causes harmful interference to radio or television reception, which can be determined by turning the equipment off or on, you are encouraged to try to correct the interference by one or more of the following measures:

- ensure attached cables do not lie across the card
- reorient the receiving antenna
- increase the distance between the equipment and the receiver
- connect the equipment into an outlet on a circuit different from that to which the receiver is connected
- consult the dealer or an experienced radio/TV technician for help

Note

It is recommended that wherever possible shielded interface cables be used.

Contents
Core Tile for ARM11 MPCore User Guide

List of Tables
Core Tile for ARM11 MPCore User Guide

List of Figures
Core Tile for ARM11 MPCore User Guide

Preface

This preface introduces the RealView Core Tile for ARM11 MPCore and its reference documentation. It contains the following sections:

- *About this book* on page xii
- *Feedback* on page xv.

About this book

This book describes how to set up and use the CT11MPCore.

Intended audience

This book has been written for experienced hardware and software developers to aid the development of ARM-based products using the CT11MPCore as part of a development system.

Using this book

This book is organized into the following chapters:

Chapter 1 ***Introduction***

This chapter introduces the CT11MPCore.

Chapter 2 ***Getting Started***

This chapter describes how to set up and prepare the CT11MPCore for use.

Chapter 3 ***CT11MPCore Hardware Description***

This chapter describes the on-board hardware of the CT11MPCore.

Chapter 4 ***Test Chip Hardware Description***

This chapter describes the features of the ARM11 MPCore test chip which affect the CT11MPCore.

Chapter 5 ***CT11MPCore Signal Descriptions***

This chapter provides a summary of signals present on the CT11MPCore connectors, test points, LED indicators, and the links that can be modified to change signal routing.

Appendix A ***Specifications***

This appendix contains the electrical, timing, and mechanical specifications for the CT11MPCore.

Typographical conventions

The following typographical conventions are used in this document:

bold Highlights ARM signal names within text, and interface elements such as menu names. This style is also used for emphasis in descriptive lists where appropriate.

italic Highlights special terminology, cross-references and citations.

`monospace` Denotes text that can be entered at the keyboard, such as commands, file names and program names, and source code.

`monospace` Denotes a permitted abbreviation for a command or option. The underlined text can be entered instead of the full command or option name.

`monospace italic` Denotes arguments to commands or functions where the argument is to be replaced by a specific value.

`monospace bold` Denotes language keywords when used outside example code.

Further reading

This following publications by ARM Limited provide additional information on using the Core Tile and related products:

- *ARM11 MPCore Processor Technical Reference Manual* (ARM DDI 0360)
- *L220 Cache Controller Technical Reference Manual* (ARM DDI 0329)
- *RealView Emulation Baseboard User Guide* (ARM DUI 0303)
- *LT-XC2V4000+ User Guide* (ARM DUI 0186B)
- *ARM RealView AT1 Analyzer Tile User Guide* (ARM DUI 0189)
- *ARM Multi-ICE User Guide* (ARM DUI 0048)
- *AMBA Specification* (ARM IHI 0011)
- *AMBA 3 AXI Protocol Specification* (ARM IHI 0022)
- *ARM Architecture Reference Manual* (ARM DDI 0100)
- *ADS Tools Guide* (ARM DUI 0067)
- *ADS Debuggers Guide* (ARM DUI 0066)
- *ADS Debug Target Guide* (ARM DUI 0058)
- *ADS Developer Guide* (ARM DUI 0056)
- *ADS CodeWarrior IDE Guide* (ARM DUI 0065)
- *RealView Debugger User Guide* (ARM DUI 0153)
- *RealView Compilers and Libraries Guide* (ARM DUI 0205)
- *RealView Linker and Utilities Guide* (ARM DUI 0206)

- *RealView ICE User Guide* (ARM DUI 0155).

Feedback

ARM Limited welcomes feedback both on the ARM Core Tiles and on the documentation.

Feedback on this document

If you have any comments about this document, send email to `errata@arm.com` giving:

- the document title
- the document number
- the page number(s) to which your comments refer
- an explanation of your comments.

General suggestions for additions and improvements are also welcome.

Feedback on the ARM Core Tiles

If you have any comments or suggestions about these products, contact your supplier giving:

- the product name
- an explanation of your comments.

Chapter 1
Introduction

This chapter introduces the Core Tiles. It contains the following sections:

- *About the Core Tiles* on page 1-2
- *Overview of CT11MPCore* on page 1-3
- *Precautions* on page 1-11.

Note

This guide covers the HBI-0146 (CT11MPCore) printed-circuit board.

This board supports the ARM11 MPCore test chip. The availability of Core Tiles however, depends on the availability of individual test chips. Contact your sales representatives for details on currently available Core Tiles.

1.1 About the Core Tiles

Core Tiles are development boards that enable you to develop products based on ARM processors and AMBA interfaces. Core Tiles are built around test chips, which are ASIC implementations of one or more ARM processors. Core Tiles provide one or more AMBA interfaces from the processor so that it can be connected to an AMBA-based system. See the *AMBA Specification* (ARM IHI 0011) and the *AMBA 3 AXI Protocol Specification* (ARM IHI 0022) for further information.

The Core Tile must be used in conjunction with a specific baseboard that implements the necessary system and memory controllers in an FPGA. The CT11MPCore is combined with the RealView Emulation Baseboard (EB) to provide a standalone system for product development. Third-party or custom development systems may also be used with a Core Tile.

Core Tiles do not have power or JTAG connectors. The tiles must be stacked on a baseboard that provides power and JTAG connections. The Core Tiles also require a reference clock (or clocks) to be supplied by the attached baseboard.

Through-board connectors on tile products allow stacking of multiple systems. Multiple combinations of Core Tile and Logic Tile can be used to create a multiprocessor system.

Note

Using the EB, you can build a dual processor system using Core Tiles without the need for Logic Tiles. Using other baseboards, for example the PB926EJ-S, you must include a Logic Tile between the baseboard and the Core Tile.

For debug, a Realview Analyzer Tile can be used to provide access to signals on the tile header connectors.

1.2 Overview of CT11MPCore

The major components on the CT11MPCore are as follows:

- microprocessor test chip
- tile headers on the top and bottom of the board
- bus interface logic
- clock selection and initialization logic
- controllable power supply for the test chip core and PLL
- test chip power monitoring logic
- configuration and control PLD
- ARM11 MPCore test chip signals routing
- AXI bus multiplexing logic.

Figure 1-1 on page 1-4 shows the layout of a CT11MPCore.

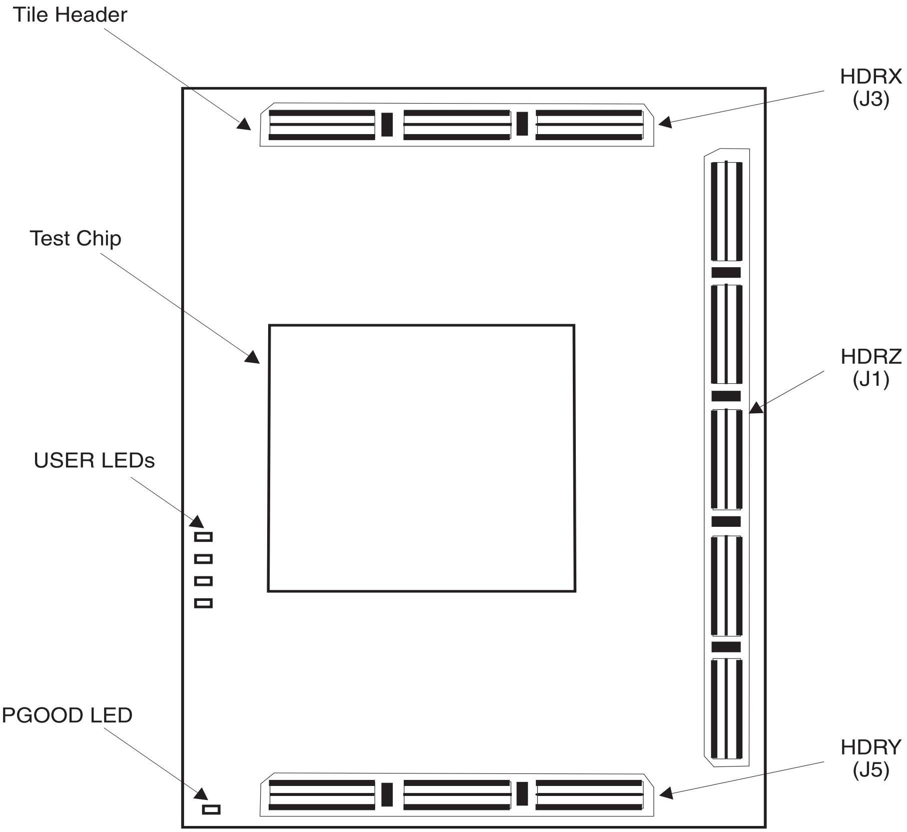


Figure 1-1 CT11MPCore layout

Figure 1-2 on page 1-5 shows a CT11MPCore together with an EB. Typically, the remaining tile site is occupied by a Core Tile or Logic Tile which can be used for custom peripheral development or to hold a synthesized core.

Refer to the *LT-XC2V4000+ Logic Tile User Guide* (ARM DUI 0186B) for details on the Logic Tile which can be used for system prototyping.

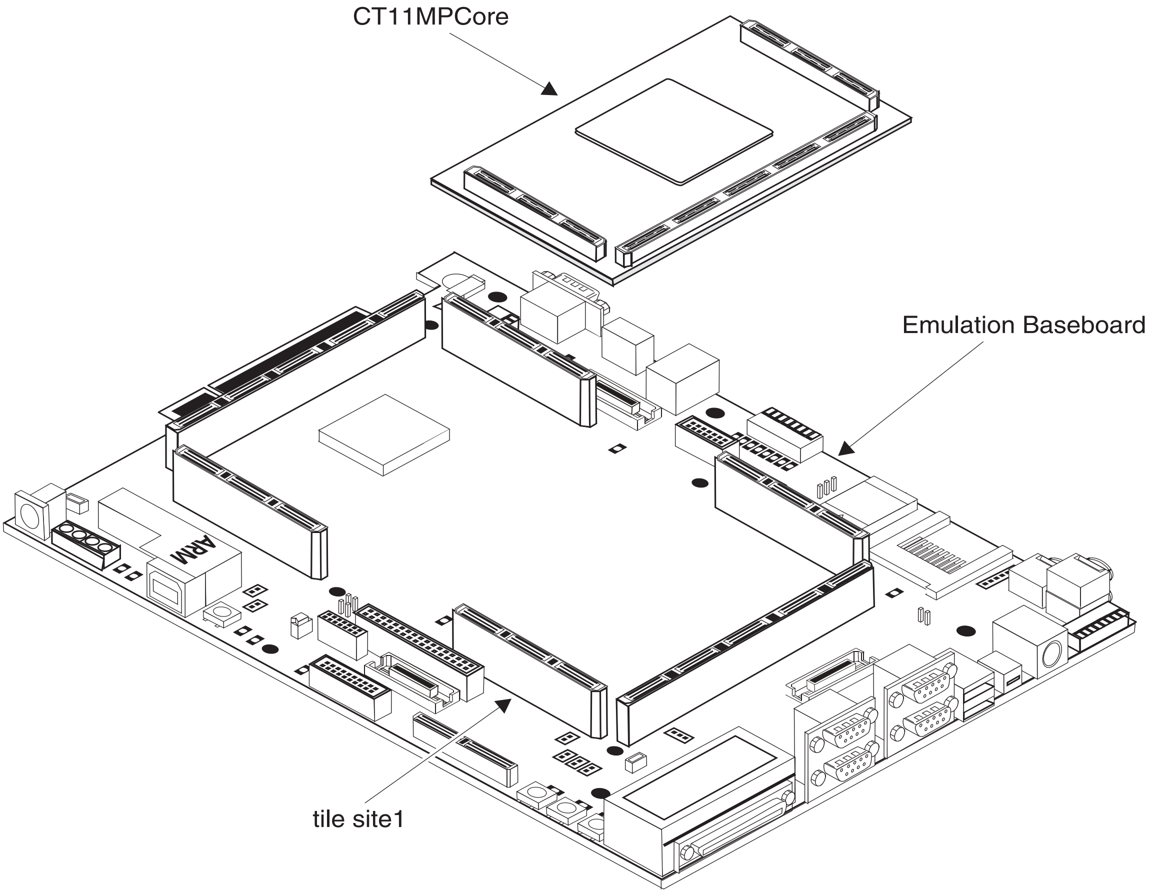


Figure 1-2 CT11MPCore and Emulation Baseboard

1.2.1 System architecture

Figure 1-3 on page 1-7 shows the architecture of a hardware development system consisting of a CT11MPCore and an EB. For details of the peripheral devices implemented on the EB and the available interfaces see *RealView Emulation Baseboard User Guide* (ARM DUI 0303).

1.2.2 External logic

The EB provides:

- power and JTAG connectors
- a reference clock
- a serial interface to the CT11MPCore PLD that loads the desired configuration
- initialization values to the PLD for the PLL and clock divider in the test chip
- peripheral devices (for example, memory controller, interrupt controller, DMA controller, system and reset controller, serial I/O). See *RealView Emulation Baseboard User Guide* (ARM DUI 0303) for details.

The external connectors and control logic are also provided by the baseboard. The FPGA on the baseboard or a Logic Tile provides system control functions for the CT11MPCore. See Chapter 3 *CT11MPCore Hardware Description*, and the documentation for your EB for more details.

Note

The FPGAs on the external baseboard or Logic Tile must be loaded with an appropriate image. An application note, giving information on the FPGA images for the combination of products that you are using may be available, please refer to the documentation provided on the product CDs.

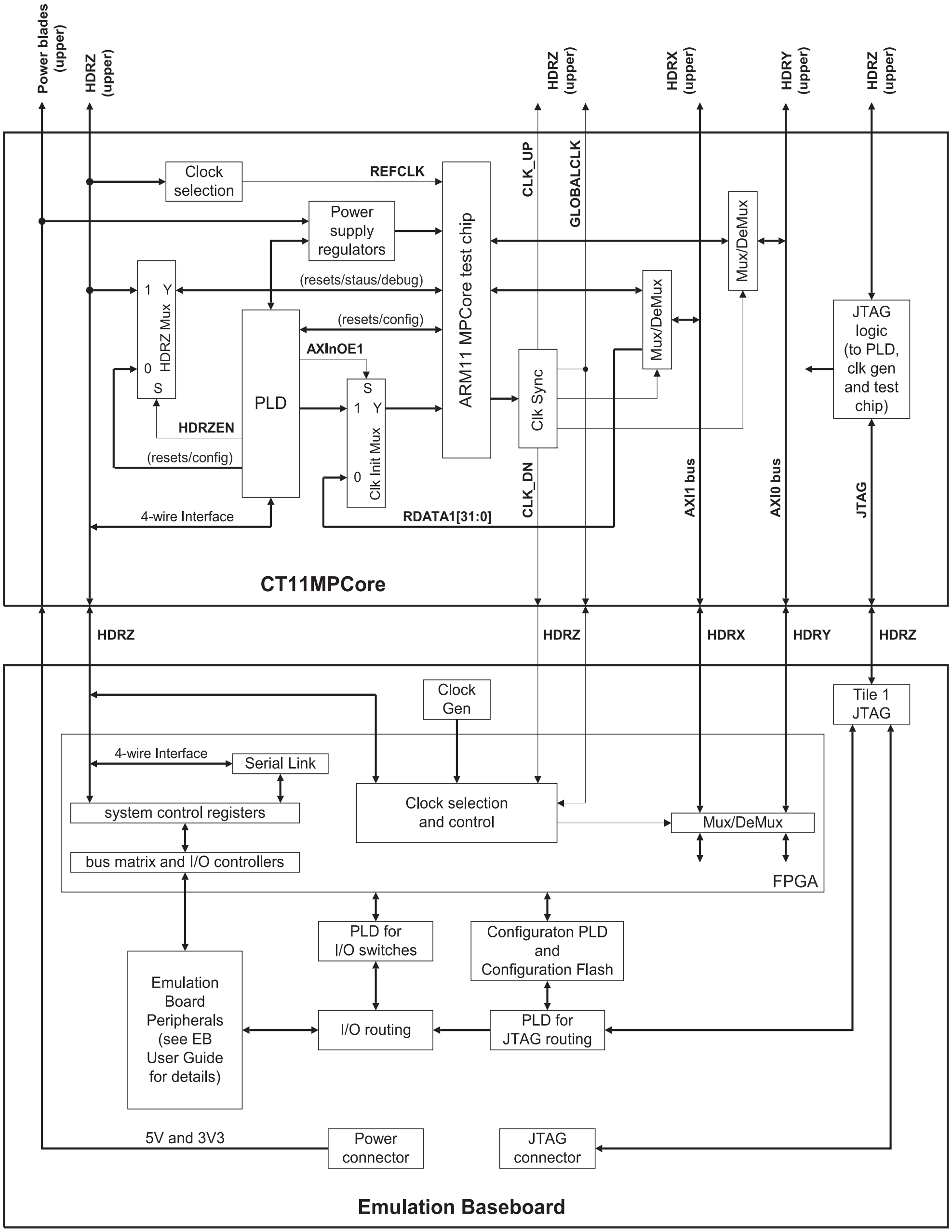


Figure 1-3 System block diagram

1.2.3 ARM processor test chip

The name of the Core Tile reflects the test chip fitted. The CT11MPCore has an ARM11 MPCore test chip which implements four ARM11 MPCore processors and a L220 level 2 cache controller.

——— **Note** ———

For more details on processors and test chips, see the *Technical Reference Manual* for the processor family. Details on variations between test chips for the same processor family are also described in documents in the `product_name\docs\test chips` directory of the product CD.

———

Depending on the test chip, the core configuration signals are driven by the PLD or the FPGA on the baseboard, or by resistor links. In a production ASIC, these core signals are permanently tied HIGH or LOW. However, you can change the PLD register values via a 4-wire serial link controlled by the system registers in the baseboard, in order to experiment with different processor configurations. See *Overview of Core Tile configuration* on page 3-36 for details.

1.2.4 PLD

The PLD on the CT11MPCore receives configuration signals from the serial link, controlled by the baseboard.

The CT11MPCore PLD outputs signals that control:

- the DACs that control the power supplies for the test chip
- the clock control registers that control the reference clock source to the core
- the dual AXI bus interface
- the debug cross-trigger matrix
- multiplexing of some of the signals on the HDRZ header
- the test chip resets and configuration signals.

The PLD transmits status signals to the serial link:

- a PLD identifier (PLDVER) that can be used to identify the version of the PLD image
- the value of the test chip voltages
- the current drawn from the test chip core and PLL
- the test chip status signals.

——— **Note** ———

Different Core Tiles might also transmit signals that are specific to that tile. For example, the CT11MPCore transmits a MPCore PLL update complete signal.

User control of the signals present in the serial interface is possible through the system control registers on the baseboard.

1.2.5 Processor bus

The test chip on the CT11MPCore has a dual AXI bus interface. See *Overview of Core Tile configuration* on page 3-36 for a description of the configuration options for the AXI bus and *Bus interface characteristics* on page A-2 for parametric data.

1.2.6 Memory

The CT11MPCore does not support PISMO Expansion Memory Modules but the ARM11 MPCore test chip has on chip memory. See *Memory configuration* on page 3-43 for details of the ARM11 MPCore test chip memory system. If additional system memory is required, you can add PISMO Expansion Memory Modules to the EB. See the *Realview Emulation Baseboard User Guide* (ARM DUI 0303) for details.

1.2.7 Clock generation

The primary reference clock for the CT11MPCore is provided by the attached EB or a Logic Tile fitted above the CT11MPCore. See *Clocks* on page 3-4 for details of the CT11MPCore clocking system.

1.2.8 JTAG

The CT11MPCore does not have a JTAG connector. You must use the JTAG connector on the EB. See *JTAG support* on page 3-60 for details of the CT11MPCore JTAG interface.

1.2.9 Power supply control

Interface logic on the CT11MPCore enables you to control and read different supply voltages. The current drawn by the test chip core and PLL can also be read to monitor power. The voltage control range available is set during board manufacture.

Power supply is by external board. The CT11MPCore requires the baseboard to provide 3V3 and 5V. The test chip core voltage and PLL voltage of 1V2 and the PLD core voltage of 1V8 are generated locally by the CT11MPCore. See *Power supply control* on page 3-28 for details of the CT11MPCore power supply interface.

1.2.10 Links and indicators

The CT11MPCore has factory-installed links for setting test chip configuration. For example, there are links to set **PERIPHBASE** which determines the base address for the ARM11 MPCore, L220 Cache Controller and ARM11 MPCore test chip specific registers. These links do not normally require changing. For more details, see *Links* on page 5-25.

The CT11MPCore has a Power Good (PGOOD) LED indicator for the VDDCORE power supply and four User LEDs. See *LED indicators* on page 5-30 for details. For details on indicators present on the Logic Tiles, Interface Modules, and baseboard products, see the documentation supplied with the product.

1.3 Precautions

This section contains safety information and advice on how to avoid damage to the Core Tile.

1.3.1 Ensuring safety

The CT11MPCore is powered from 3V3 and 5V supplies provided by the baseboard.

Warning

To avoid a safety hazard, only *Safety Extra Low Voltage* (SELV) equipment must be connected to the Core Tile.

The test chip on the Core Tile can become very hot during continuous I/O activity.

Warning

Do not touch the test chip package if the Core Tile is powered up or until its has reached a safe temperature after the Core Tile is powered down.

1.3.2 Preventing damage

The Core Tile is intended for use within a laboratory or engineering development environment. It is supplied without an enclosure and this leaves the board sensitive to electrostatic discharges and allows electromagnetic emissions.

Caution

To avoid damage to the Core Tile you must observe the following precautions:

- never subject the tile to high electrostatic potentials
- always wear a grounding strap when handling the tile
- only hold the tile by the edges
- avoid touching the component pins or any other metallic element.

Do not use the board near equipment that could be:

- sensitive to electromagnetic emissions, such as medical equipment
- a transmitter of electromagnetic emissions.

Chapter 2
Getting Started

This chapter describes how to set up and prepare the CT11MPCore for use. It contains the following sections:

- *Using the CT11MPCore with an Emulation Baseboard* on page 2-2
- *Using the CT11MPCore with a custom baseboard* on page 2-4
- *Connecting power* on page 2-5
- *Connecting RealView ICE* on page 2-7.

——— **Note** ———

The information in this chapter provides only a general overview. Details on using a Core Tile with different ARM products are covered in application notes. Ensure that you use the correct application note and FPGA image for your configuration.

The CT11MPCore must be used with a baseboard that provides the configuration control and additional peripherals. This requirement is fully met by the EB. The FPGA on the baseboard must contain an image that is explicitly designed to interface with the CT11MPCore.

———————————

2.1 Using the CT11MPCore with an Emulation Baseboard

A typical multi-processor development system is shown in Figure 2-1 on page 2-3.

To set up a development system using the *Emulation Baseboard* (EB):

1. Fit the CT11MPCore to tile site 1 on the EB.

2. If required, fit a further Core Tile or a Logic Tile to tile site 2 on the EB.

3. If required you can connect additional sets of Logic Tile and Core Tile at either tile site to create a tile stack.

   Note

   To build a tile stack above the CT11MPCore, AXI bus multiplexing and demultiplexing logic must be included on the Logic Tile above the CT11MPCore. This will reduce the maximum operating frequency of tile site 1. It is recommended, for systems requiring additional processors, that the processor tile stack is built at tile site 2 to maximize system performance.

4. If required, you can connect a RealView Analyzer Tile between the baseboard and the Core Tile, to enable monitoring of signals using a logic analyser.

5. You can also place a Logic Tile and an Interface Tile on the top of the tile stack. The Logic Tile can be loaded with an appropriate image that contains your peripherals and the connectors on the Interface Tile can provide access to your peripherals instantiated in the Logic Tile.

6. Connect a JTAG debugger to the baseboard (see *Connecting a JTAG device to the EB* on page 2-8).

7. Set the CONFIG slide-switch on the baseboard to ON.

8. Supply power to the EB (see *Supplying power to the EB* on page 2-5).

9. Load the appropriate FPGA images into the Logic Tiles. See the application note for details on the image to use and see the *RealView Emulation Baseboard User Guide* and *LT-XC2V4000+ User Guide* for details on programming procedures.

10. Set the CONFIG slide-switch to OFF, power cycle the board, and load your application program.

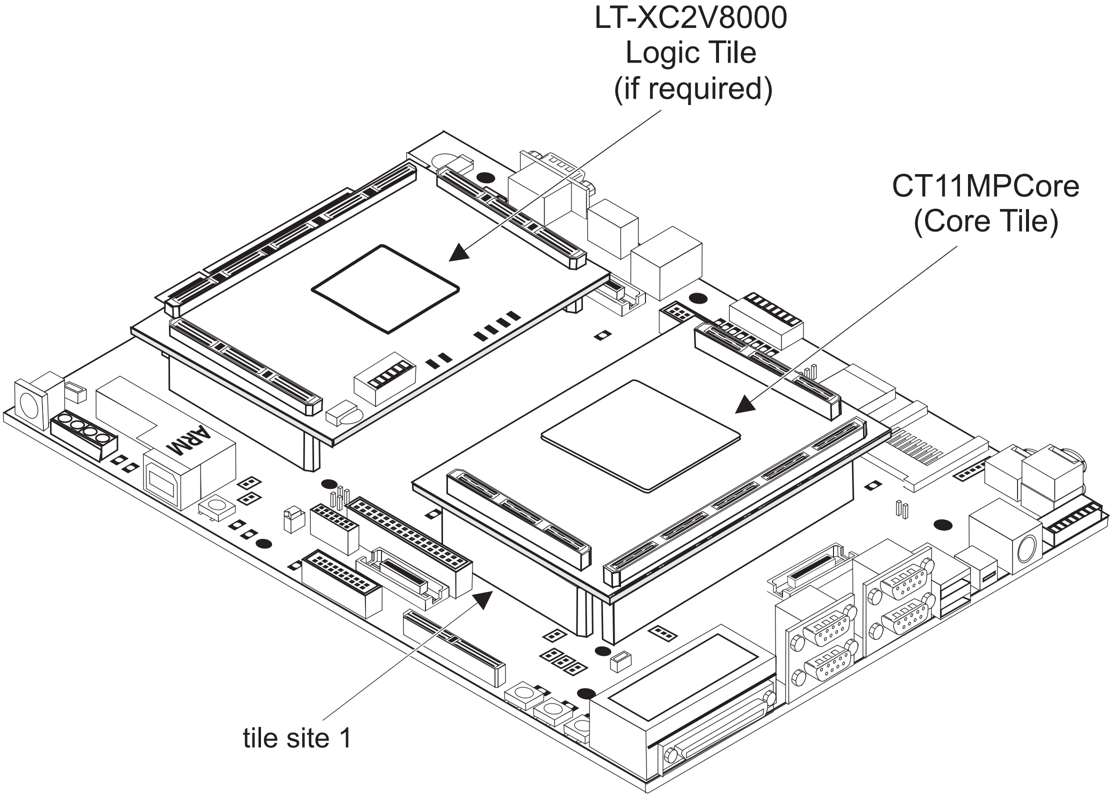


Figure 2-1 Core Tile, Logic Tile, and an Emulation Baseboard

Note

For details on how to load and run applications on an Emulation Baseboard system, see the *RealView Emulation Baseboard User Guide* (ARM DUI0303).

2.2 Using the CT11MPCore with a custom baseboard

If you are designing a custom baseboard to accept a CT11MPCore, you must ensure that your board meets the following requirements:

Mechanical layout

The mechanical layout is described in Appendix A *Specifications*.

Power supplies

The CT11MPCore uses 5V and 3V3 supplies connected through power-blades in the header connectors. See *Header connectors* on page 5-2 for details. The CT11MPCore uses the 5V supply to locally generate the 1V8 supply for the PLD core, the 1V2 supply for the test chip core (VDDCORE), and the 1V2 supply for the test chip PLL (AVDD). The 3V3 supply is used by the internal I/O and external interfaces. The test chip core and PLL voltage are controlled by logic in the PLD on the CT11MPCore. See *Power supply control* on page 3-28 for details on the voltage control logic.

Clock control

Your primary reference clock must be supplied by an attached Logic Tile or baseboard. The CT11MPCore clocking system is described in Chapter 3 *CT11MPCore Hardware Description* and the ARM11 MPCore test chip specific clocking requirements are described in Chapter 4 *Test Chip Hardware Description*. See also the *ARM11 MPCore Processor Technical Reference Manual* (ARM DDI 0360).

JTAG control

The CT11MPCore does not have a JTAG connector. The baseboard must provide a JTAG connector and route the JTAG signals to the header signals on the Core Tile. JTAG routing is described in *JTAG support* on page 3-60 and *Debug and JTAG configuration* on page 4-31.

Bus configuration signals

The ARM11 MPCore dual AXI bus and some of the signals on header HDRZ are controlled through multiplexers. Bus configuration is managed by the PLD on the Core Tile. See *Overview of Core Tile configuration* on page 3-36 for details.

AXI slaves

In order for the ARM11 MPCore to boot, the baseboard must implement AXI slaves that cover the 4GB memory map.

2.3 Connecting power

CT11MPCore power is supplied through power-blades in the header connectors. The power source is connected to the EB.

2.3.1 Supplying power to the EB

If the EB is not inserted into a PCI enclosure, you must connect the supplied 12VDC brick power supply to power socket J28 or an external bench power supply to the screw-terminal connector J27. See Figure 2-2.

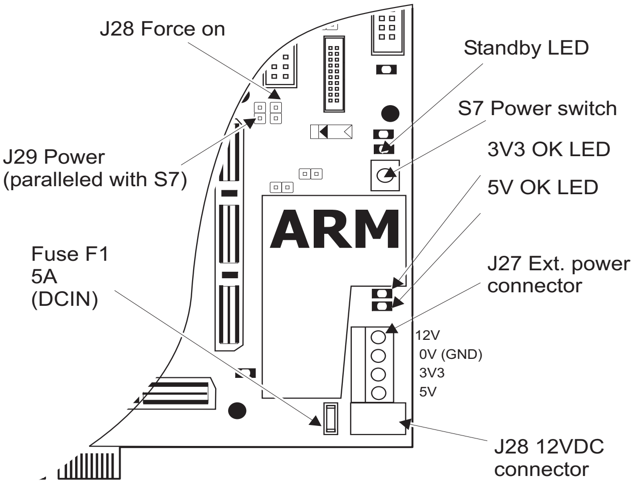


Figure 2-2 EB power connectors

Note

If you are using the supplied brick power supply connected to J28, the Standby/power push button toggles the power on and off.

If you are using an external power supply connected to J27, or you are powering the board from the PCI backplane, the Standby/power switch is not used and power is controlled by shutting down the external power source.

Caution

You must only use one power source for the system. Use only the PCI connector, J27, or J28, do not, for example, use the PCI connector and J27 at the same time.

If you are using separate 5V and 3V3 bench supplies, always set sensible current limits. The 3V3 supply should track the 5V supply. On the EB, the 3V3 supply tracks the 5V supply after an initial 1ms delay.

2.4 Connecting RealView ICE

The Core Tile does not have a JTAG connector, but there are JTAG signals present on the header connectors. The Core Tile must be used with an EB or custom baseboard that contains a JTAG connector. External JTAG equipment can be used to:

- Download and debug programs.

  You can connect RealView ICE (RVI) or other JTAG debuggers to the external JTAG connector and download and debug programs.

- Download new images to the PLD on the Core Tile or to FPGAs or PLDs present on other attached boards.

  For a CT11MPCore attached to an EB, either Multi-ICE, RVI or the USB debug port on the EB can be used for downloading FPGA images, but only RVI can be used for debug.

Selection between debugging programs and downloading new images to the FPGA is controlled by the CONFIG slide-switch on the EB. See the documentation supplied with the EB for more details on connecting JTAG and using the CONFIG slide-switch.

Caution

Because the Core Tile does not provide nonvolatile memory, programs are lost if the power is removed. Use flash memory for nonvolatile storage.

The flash memory can be:

- any unused space in the EB NOR flash
- nonvolatile memory available in the PISMO memory expansion slots on the EB
- DiskOnChip flash on the EB.

Do not use spare EB configuration flash space (`address range: 0x4C000000–0x4DFFFFFF`) for program storage.

See the *Programmer's Reference* chapter in the *RealView Emulation Baseboard User Guide* (DUI0303) for details of the EB memory map.

The JTAG connector provides a set of signals that allow debugging equipment to be used. If you are debugging a development system with multiple tiles, connect the JTAG debugging equipment to the EB and the JTAG signals will be routed through any connected tiles. See *JTAG loopback control* on page 3-68 for details of the JTAG routing within a tile stack.

The JTAG debug and configuration paths are described in *JTAG support* on page 3-60 and *Debug and JTAG configuration* on page 4-31.

2.4.1 Connecting a JTAG device to the EB

The JTAG setup for the EB and RVI is shown in Figure 2-3. (Refer to the documentation supplied with your debugger for information on connecting other JTAG interface products.)

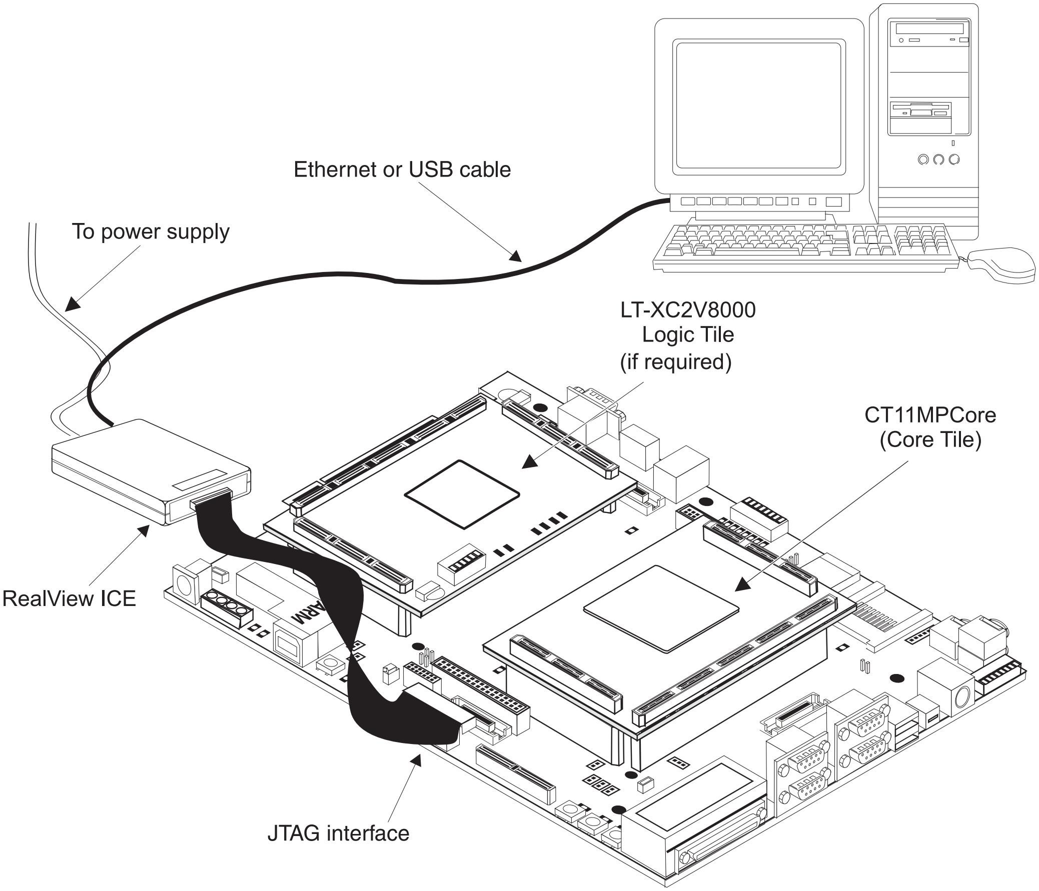


Figure 2-3 JTAG connection to an Emulation Baseboard

Note

The ARM11 MPCore test chip does not support trace.

Chapter 3
CT11MPCore Hardware Description

This chapter describes the on-board hardware of the CT11MPCore. It contains the following sections:

Note

The HBI-0146 board supports the ARM11 MPCore test chip, but the availability of boards depends on the availability of test chips. Contact your sales representatives for details on currently available Core Tiles.

3.1 Core Tile architecture

The high level architecture of the CT11MPCore is shown in Figure 3-1.

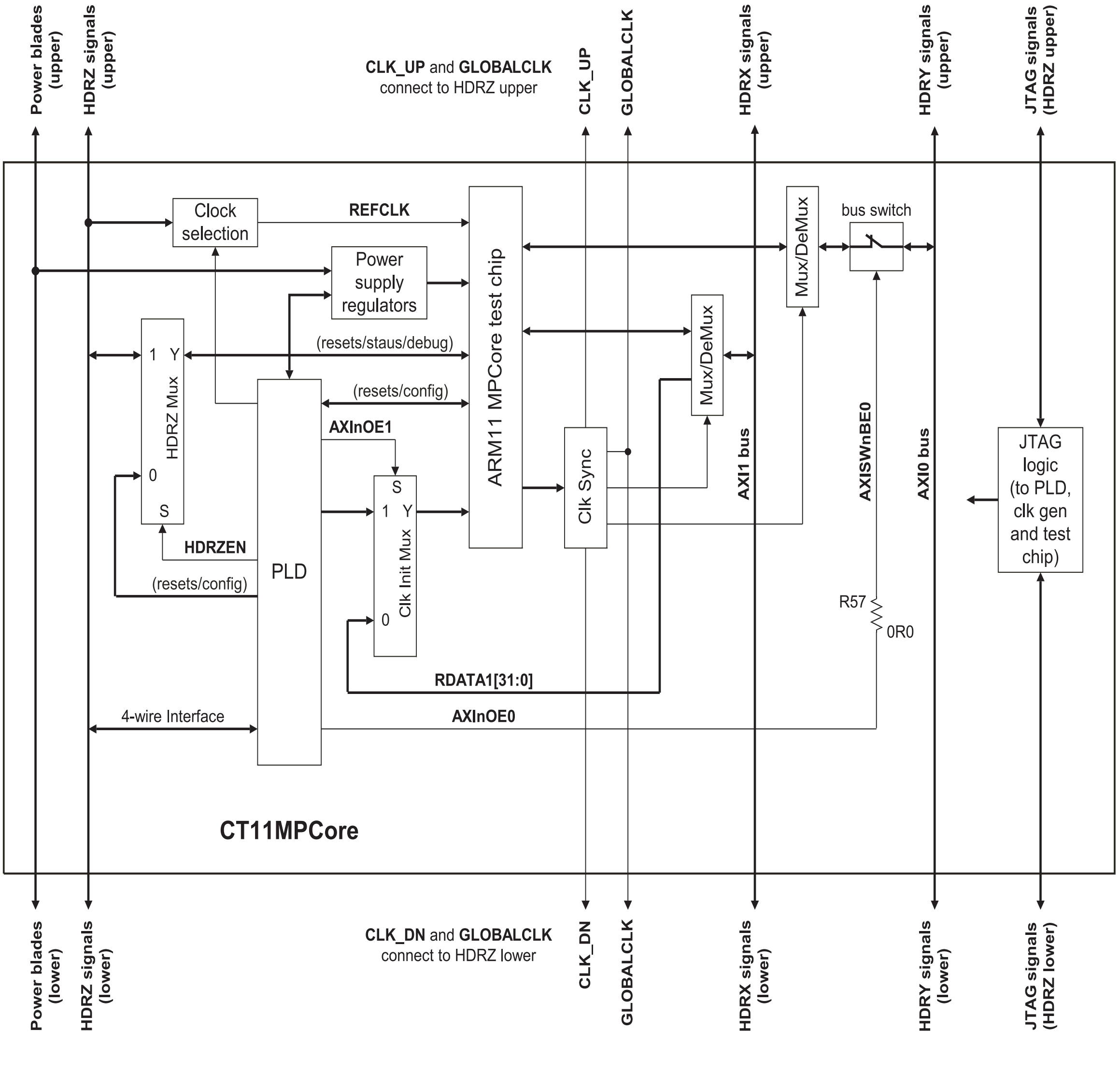


Figure 3-1 CT11MPCore block diagram

3.2 About the ARM11 MPCore test chip

The ARM11 MPCore test chip is a proof-of-concept vehicle for the MPCore macrocell and is built for two reasons:

- It enables you to functionally validate the ARM11 MPCore macrocell on the silicon process it is built on.
- You can use it as a building block to create prototype systems designed for product and operating system development.

The ARM11 MPCore test chip functionality consists of:

- four ARM11 MPCore CPUs, each CPU may be suspended (clock is stopped)
- L220 level 2 cache controller
- PLL
- boundary scan logic
- input and output pads.

Note

For more details see Chapter 4 *Test Chip Hardware Description* on page 4-1 and the *ARM11 MPCore Processor Technical Reference Manual* (ARM DDI 0360).

3.3 Clocks

The CT11MPCore does not have an on board reference clock. The single input clock required by the test chip PLL, **REFCLK** is routed either from the board below as **CLK_NEG_UP_IN**, or the board above as **CLK_NEG_DN_IN**, depending on the position of the CT11MPCore in the system stack and the board combination used.

As shown in Figure 3-2 on page 3-5 clock distribution and phase control of the AXI clock output from the test chip is provided on board by an ispClock5620 programmable clock generator. The ispClock5620 is PLL based and provides independent programmable control of termination, slew-rate, and timing skew. All configuration information is stored on-chip in nonvolatile EEPROM memory. The CT11MPCore uses the ispClock5620 in a single pre-programmed standalone configuration.

Note

Figure 3-2 on page 3-5 is a simplified clock distribution diagram. The CT11MPCore uses two ispClock5620 programmable clock generators, one driven from **CLKOUTDIV** and one driven from **CLKOUTDIVD**. The additional de-skewed clock sources are required to multiplex the AXI port 0 and AXI port1 signals from the test chip onto the CT11MPCore X, Y and Z headers. See *AXI bus multiplexing* on page 3-31 for details.

3.3.1 Clock routing

The clock routing for the CT11MPCore and EB combination is shown in Figure 3-2 on page 3-5. This figure also shows the clock hardware initialization signals.

Selection of the clock source for the Core Tile is controlled by **CSEL** from the CT11MPCore PLD. **CSEL** selects the clock source as **CLK_NEG_DN_IN** from the board above, or **CLK_NEG_UP_IN** from the board below. The value of **CSEL** is set locally during power up by the CT11MPCore PLD.

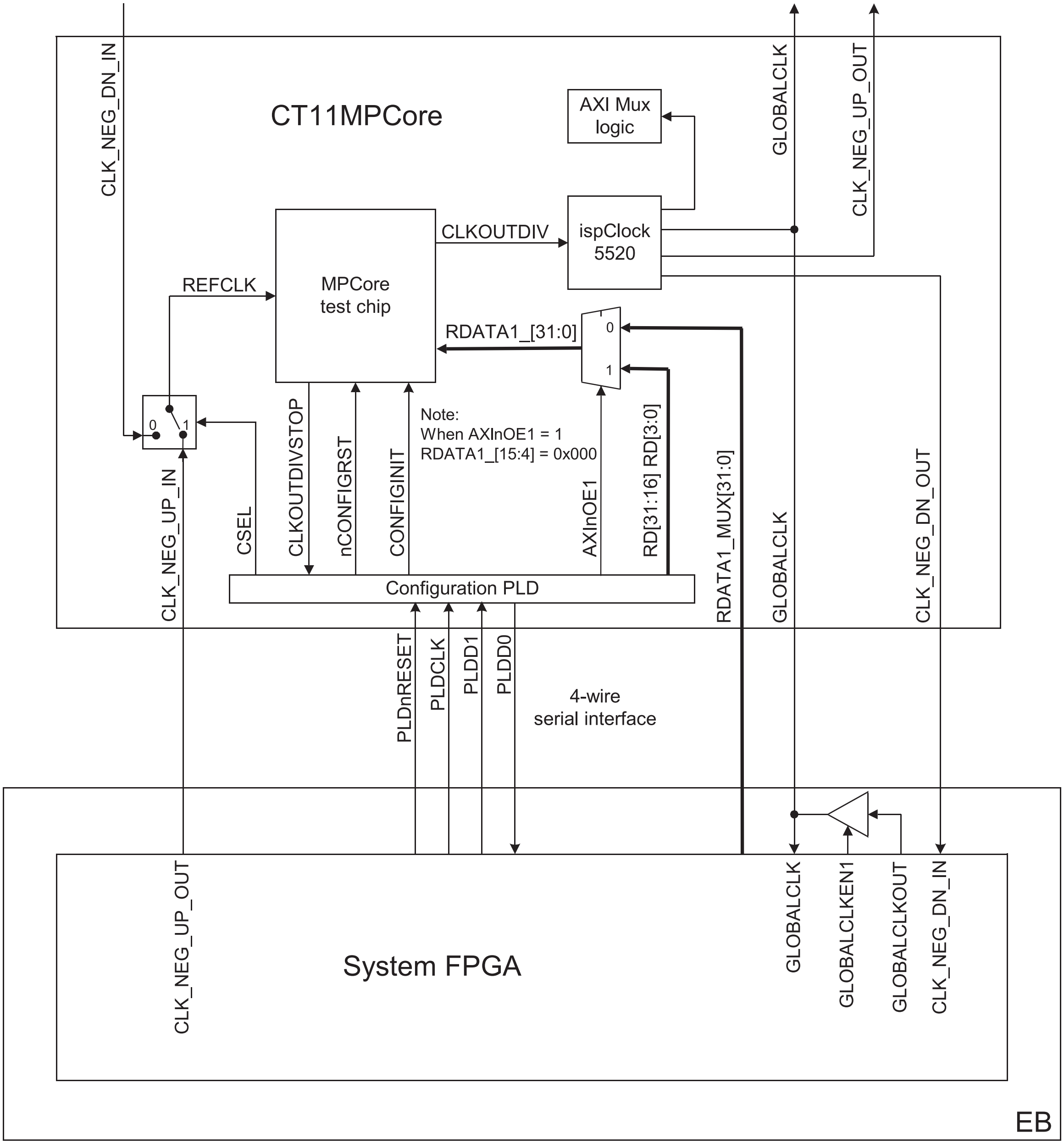


Figure 3-2 Clock routing and hardware initialization

3.3.2 Clock setting in reset mode

The **CONFIGINIT**, **nCONFIGRST**, **RD[31:16]** and **RD[3:0]** signals initialize the *test chip clock divider* and *test chip PLL control* registers during reset.

——— **Note** ———

Hardware initialization of the PLL and clock divider is controlled by firmware on the system baseboard. Hardware initialization values are transferred during reset to the PLD on the CT11MPCore via a 4-wire serial interface, see *Clock routing and hardware initialization* on page 3-5 for details.

If you only require changing the core to AXI bus clock ratio, and can tolerate resetting the CT11MPCore, you can use the simpler procedure detailed in *Changing the core to AXI bus clock ratio* on page 3-55.

———————————

The **CONFIGINIT** and **nCONFIGRST** signals are controlled by the CT11MPCore PLD and are not accessible at the Core Tile headers.

nCONFIGRST

Resets the *test chip clock divider* and *test chip PLL control* registers.

CONFIGINIT

Enables clocking of the *test chip clock divider* and *test chip PLL control* registers. The **CONFIGINIT** pulse defines valid data on the **RDATA1[31:0]** bus and is HIGH for at least two **REFCLK** cycles.

——— **Note** ———

CONFIGINIT enables simultaneous clocking of both registers. Any write access to the *test chip clock divider* register also writes to the *test chip PLL control* register and the other way round. To prevent corruption of data in either register, both **RDATA1[31:16]** and **RDATA[3:0]** are valid when **CONFIGINIT** is HIGH.

———————————

The **RD[31:16]** and **RD[3:0]** values are driven by the CT11MPCore PLD.

RD[31:16]

When **CONFIGINIT** is high, drives bits [15:0] of the test chip *PLL control register* via **RDATA1[31:16]** on the test chip.

RD[3:0]

When **CONFIGINIT** is high, drives bits [3:0] of the test chip *Clock divider register* via **RDATA1[3:0]** on the test chip.

Figure 3-3 shows the **RDATA1[31:0]** data fields when using **CONFIGINIT**.

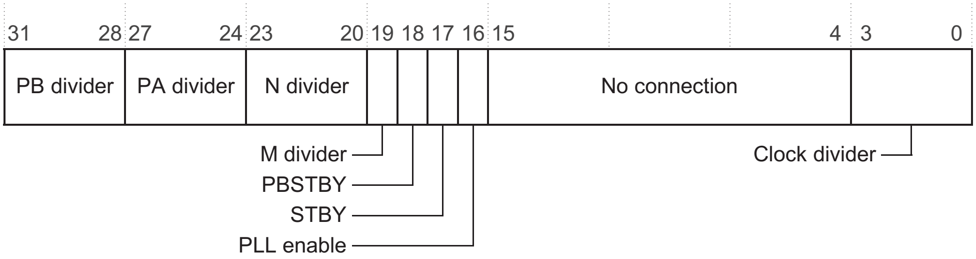


Figure 3-3 RDATA1[31:0] data fields when using CONFIGINIT

The control sequence for loading the clock divider setting in reset mode is shown in Figure 3-4 on page 3-8. In this example, the value written to the *Clock divider register* is b0001, setting it to divide by two. Since the *PLL control register* is also written to at this time, the full value written to **RDATA1[31:0]** is `0x00060001` so that:

- PLLEN= 0 masking the software PLL enable
- STBY=1 setting **CLKA** from the PLL to 0
- PSTBY=1 setting **CLKB** from the PLL to 0.

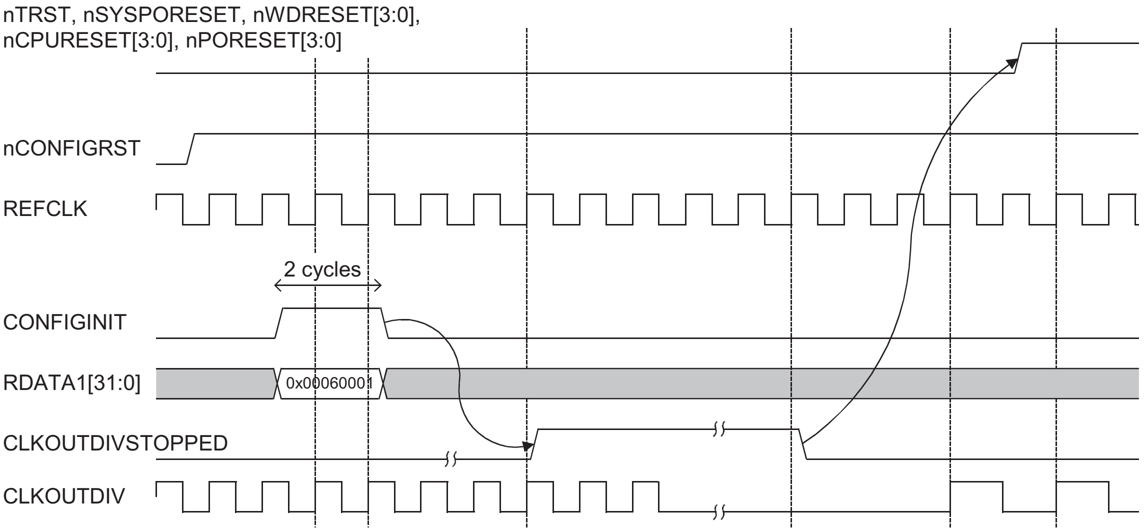


Figure 3-4 Clock divider setting in reset mode, dividing by two

The control sequence for loading the *PLL control register* is shown in Figure 3-5 on page 3-10. In this example, the PLL is set to multiply by four.

The initial conditions are:

- PLLEN=0 masking the software PLL enable
- STBY=1 setting **CLKA** from the PLL to 0
- PSTBY=1 setting **CLKB** from the PLL to 0
- clock divider=b0001 setting to divide by two.

The following values are driven on **RDATA1[31:0]**:

Note

- **STBY** must remain HIGH at least 15µs after Vdd becomes stable
- **REFCLK** must have been running for at least 1µs before **STBY** is released.

`0x01160001` Set the PLL divider values in PLL standby mode:

- PB divider=0
- PA divider=1

- N divider=1
- M=0.

PBSTBY and **STBY** remain HIGH.

The clock divider remains b0001.

`0x01140001` Release the **STBY** bit.

STBY=0 and **PBSTBY** remains HIGH.

PB divider, PA divider, N divider, and M remain at the same values

The clock divider remains b0001.

——— **Note** ———

PBSTBY must be kept HIGH for at least 20μs after **STBY** is released.

———————————

`0x01100001` Release the **PBSTBY** bit.

PBSTBY=0 and **STBY** remains LOW.

PB divider, PA divider, N divider, and M remain at the same values.

The clock divider remains at b0001.

Wait until **PLOCK** from the PLL is HIGH.

`0x01110001` Set the PLL Enable bit.

PBSTBY=0 and **STBY**=0.

PB divider, PA divider, N divider, and M remain at the same values as for `0x01160001`.

The clock divider remains at b0001.

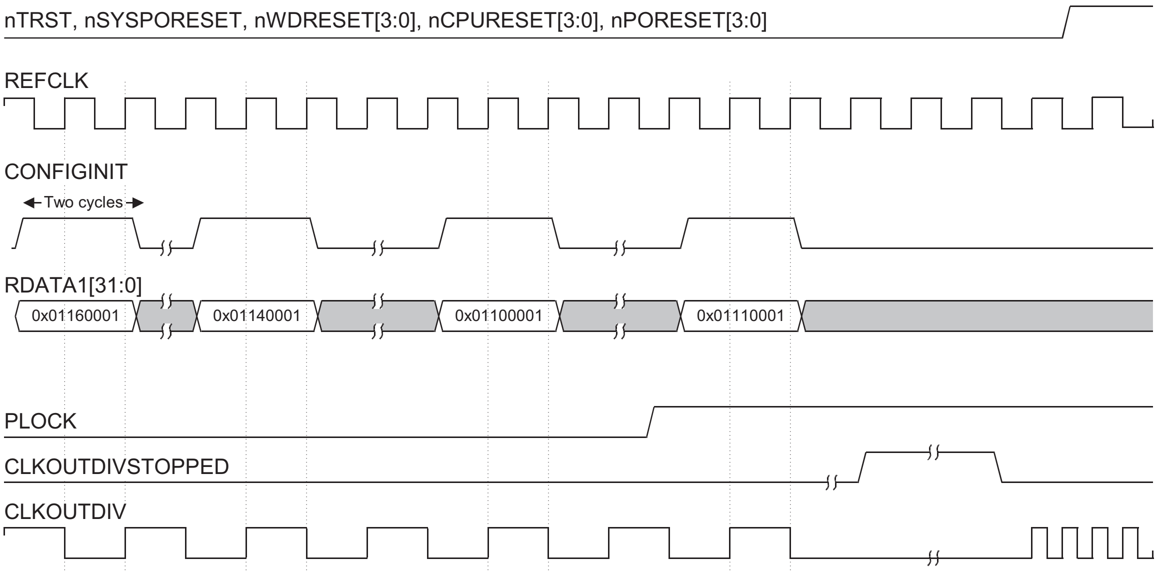


Figure 3-5 PLL setting in reset mode, multiplying by four

3.4 Resets and interrupts

This section describes the reset and interrupt signals and contains the following sections:

- *Resets*
- *Interrupts* on page 3-18.

3.4.1 Resets

This section describes the resets and contains the following subsections:

- *Baseboard resets*
- *ARM11 MPCore test chip resets*
- *Reset routing* on page 3-12
- *Resets on HDRZ signal mux* on page 3-15
- *Reset sequence* on page 3-15.

Several resets are required by the baseboard and the ARM11 MPCore test chip.

Baseboard resets

The reset logic on the baseboard initializes attached Logic Tiles and Core Tiles, the system FPGA, and external controllers as a result of a reset. The baseboard has several reset sources and generates several reset signals.

The EB reset sources and the function of the EB reset signals are described in the *RealView Emulation Baseboard User Guide* (ARM DUI0303).

The EB generates the **nCOLDRSTIN** and **nWARMRSTIN** signals for the CT11MPCore. These signals are driven by the reset controller **nSYSRST** signal within the EB system FPGA.

ARM11 MPCore test chip resets

The ARM11 MPCore test chip requires the following reset signals:

- **nSYSPORESET**
- **nCONFIGRST**
- **nWDRESET[3:0]**
- **nCPURESET[3:0]**
- **nPORESET[3:0]**

- **nTRST**.

.Chapter 4 *Test Chip Hardware Description*, in *Resets* on page 4-14.

Reset routing

The reset routing between the baseboard, CT11MPCore logic, and the ARM11 MPCore test chip is shown in Figure 3-6 on page 3-14.

In default operation, several of the ARM11 MPCore test chip resets are grouped and controlled by the CT11MPCore PLD.

If you require individual control of the ARM11 MPCore test chip resets, they can be isolated from the CT11MPCore PLD and driven from header HDRZ, see *Resets on HDRZ signal mux* on page 3-15 for details.

nCOLDRST

This reset signal from the CT11MPCore PLD controls these ARM11 MPCore test chip resets as a group:

- **nPORESET[3:0]**
- **nWDRESET[3:0]**
- **nSYSPORESET**

.

.These resets can also be controlled individually from header HDRZ by using the HDRZ signal mux.

nCPURESET[3:0]

These resets to the ARM11 MPCore test chip are controlled as a group by the **nWARMRSTIN** reset from the system FPGA. They can also be controlled individually using the serial write data register in the CT11MPCore PLD, or from header HDRZ by using the HDRZ signal mux.

- to control the **nCPURESET[3:0]** signals individually using the serial write data register in the CT11MPCore PLD, set the appropriate values in the `nCPURESET[3:0]` field in the SYS_PLD_CTRL1 register in the system FPGA on the EB
- to control the **nCPURESET[3:0]** signals individually using the HDRZ signal mux, set the appropriate value in the `HDRZEN` field in the SYS_PLD_CTRL1 register in the system FPGA on the EB.

Note

The HDRZ signal mux also enables other ARM11 MPCore test chip signals to be individually accessed via header HDRZ. See Table 3-3 on page 3-25 for a list of all the available signals.

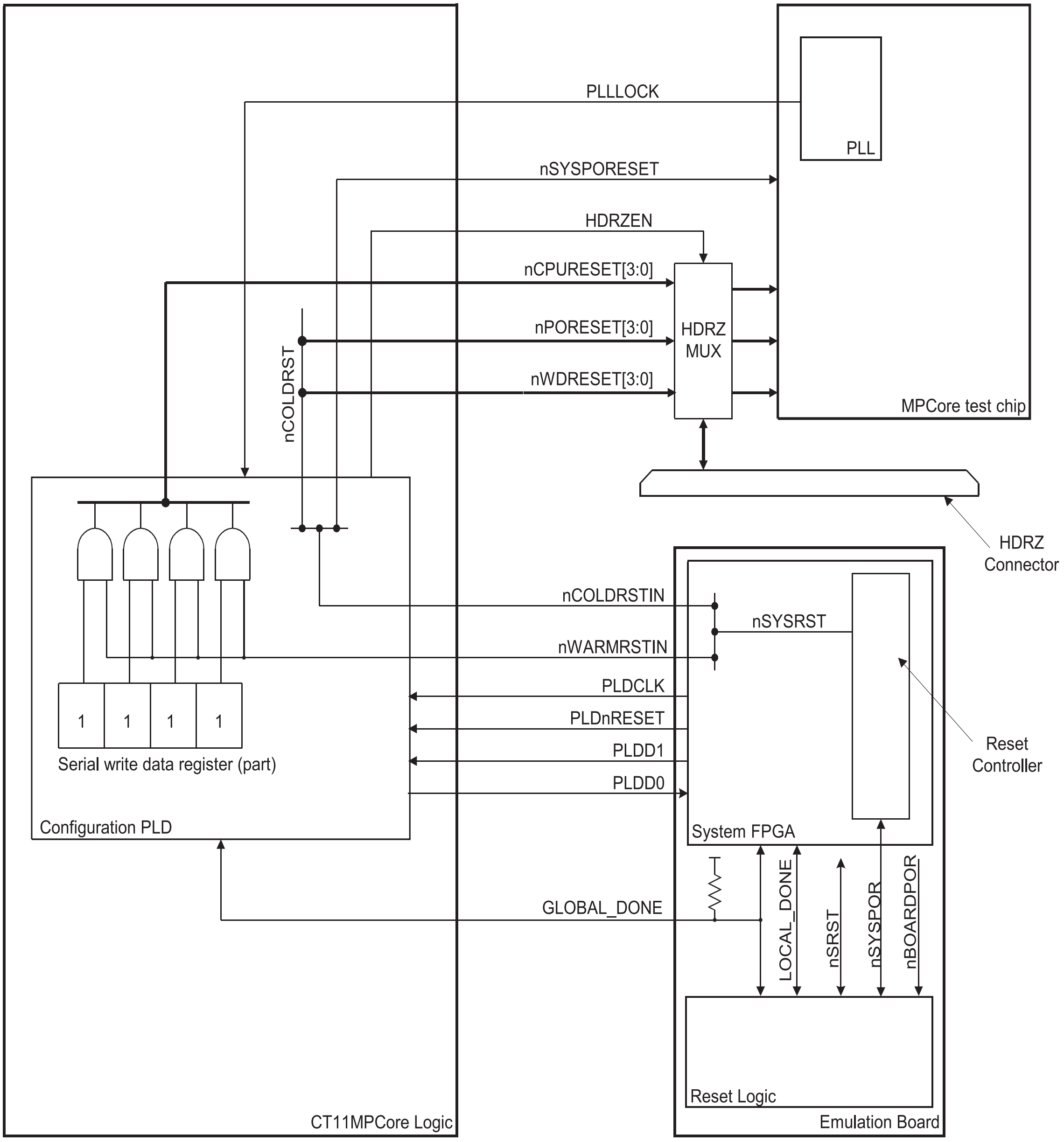


Figure 3-6 Reset routing

Resets on HDRZ signal mux

The positioning of the HDRZ signal mux is shown in *Reset routing* on page 3-14. This mux enables the resets for the ARM11 MPCore test chip to be driven individually from header HDRZ rather than as a group from the CT11MPCore PLD.

The HDRZ mux is controlled by **HRDZEN** from the CT11MPCore PLD. **HRDZEN** is controlled by writing to the `HRDZEN` field in the SYS_PLD_CTRL1 register in the system FPGA on the EB. The value written is transferred via the 4-wire serial interface to the serial write data register in the CT11MPCore PLD by continuous update.

Note

The HDRZ signal mux also enables other ARM11 MPCore test chip signals to be individually accessed via header HDRZ. Driving signals directly from this header has the principal advantage that it does not introduce any significant system delays. Controlling signals from the CT11MPCore PLD typically introduces a 2µS delay due to the serial communications link.

The HDRZ signal mux is shown in more detail in *HDRZ signal mux* on page 3-24. The ARM11 MPCore test chip signals that are available via header Z are shown in *HDRZ signal mux connections* on page 3-25.

Reset sequence

Correct initialization of a core tile and its associated baseboard requires a timed reset sequence. Details of the EB reset sequence are detailed in the *RealView Emulation Baseboard User Guide (ARM DUI 0303)*. The reset timing when using a CT11MPCore in combination with an EB is shown in *EB and CT11MPCore reset sequence* on page 3-16. The reset signals are described in Table 3-1 on page 3-17.

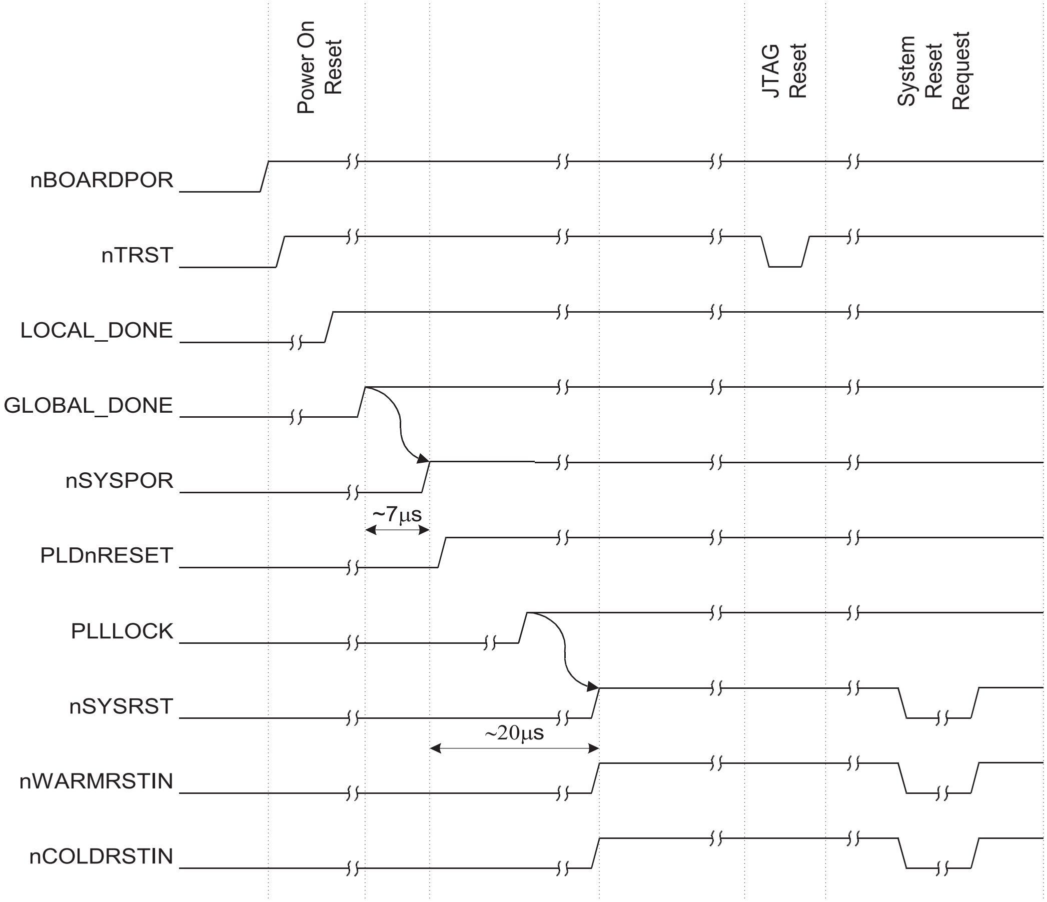


Figure 3-7 EB and CT11MPCore reset sequence

Reset signals

Table 3-1 describes the reset signals. A reset controller, implemented in the EB system FPGA, monitors the reset sources and generates the appropriate reset signals.

Table 3-1 Reset signals

Name	Function
nBOARDPOR	This signal resets the EB PLD. This signal is also used to generate **nTRST** at power on.
nTRST	Open-collector TAP controller reset (the baseboard drives this signal with **nBOARDPOR**.) **Note** **nTRST** splits into **D_nTRST**, and **C_nTRST** to provide separate debug and configuration signals on HDRZ of the CT11MPCore.
LOCAL_DONE	A configuration signal, active during reset. This signal goes HIGH when the EB system FPGA has finished configuring. **GLOBAL_DONE** is LOW until this signal goes HIGH.
GLOBAL_DONE	A configuration signal, active during reset. This is an open-collector configuration signal that goes HIGH when all FPGAs have finished configuring. The system is held in reset until this signal goes HIGH.
nSYSPOR	Power-on reset signal that initializes the reset level state machine in the EB system FPGA. There is a fixed 7µS delay after **GLOBAL_DONE** goes HIGH before **nSYSPOR** is released.
PLDnRESET	Resets the 4-wire serial interface and signals the start of each serial transfer to the CT11MPCore PLD.
PLLLOCK	A PLL status signal. **PLLLOCK** indicates when the ARM11 MPCore test chip PLL is locked to **REFCLK**. It is used by the reset controller to determine when clocks are stable and the main system reset **nSYSRST** can be released.

Table 3-1 Reset signals (continued)

Name	Function
nSYSRST	Main core tile system reset that generates **nWARMRSTIN** and **nCOLDRSTIN** to the CT11MPCore PLD. Goes HIGH approximately 20µS after **nSYSPOR** goes high.
nWARMRSTIN	Resets all CPUs in the ARM11 MPCore test chip by forcing all the **nCPURESET[3:0]** signals LOW to the ARM11 MPCore test chip.
nCOLDRSTIN	Resets the ARM11 MPCore test chip logic, all CPUs, watchdog timers and the debug logic. **Note** By default, **nCOLDRSTIN** and **nWARMRSTIN** are both driven by **nSYSRST** in the EB system FPGA. These signals act as a *Power-On-Reset* to the ARM11 MPCore test chip resetting all CPUs, the *Snoop Control Unit* (SCU), the *Interrupt Distributor*, and all ARM11 MPCore test chip logic. See *Reset handling* on page 4-15 for further details on the ARM11 MPCore test chip reset options.

3.4.2 Interrupts

This section describes the interrupt modes and contains the following subsections:

- *Interrupt routing*
- *Interrupt routing options* on page 3-21

Three interrupt modes are supported by the CT11MPCore, these are:

- Legacy mode
- Normal mode with DCC interrupt routing
- Normal mode without DCC interrupt routing

To set the interrupt mode in the CT11MPCore PLD, set the appropriate values in the `INTMODE[2:0]` field in the SYS_PLD_CTL1 register in the EB system FPGA. See *EB system FPGA registers* on page 3-46 for register details.

Interrupt routing

The interrupt routing available in the different modes is shown in Figure 3-8 on page 3-19.

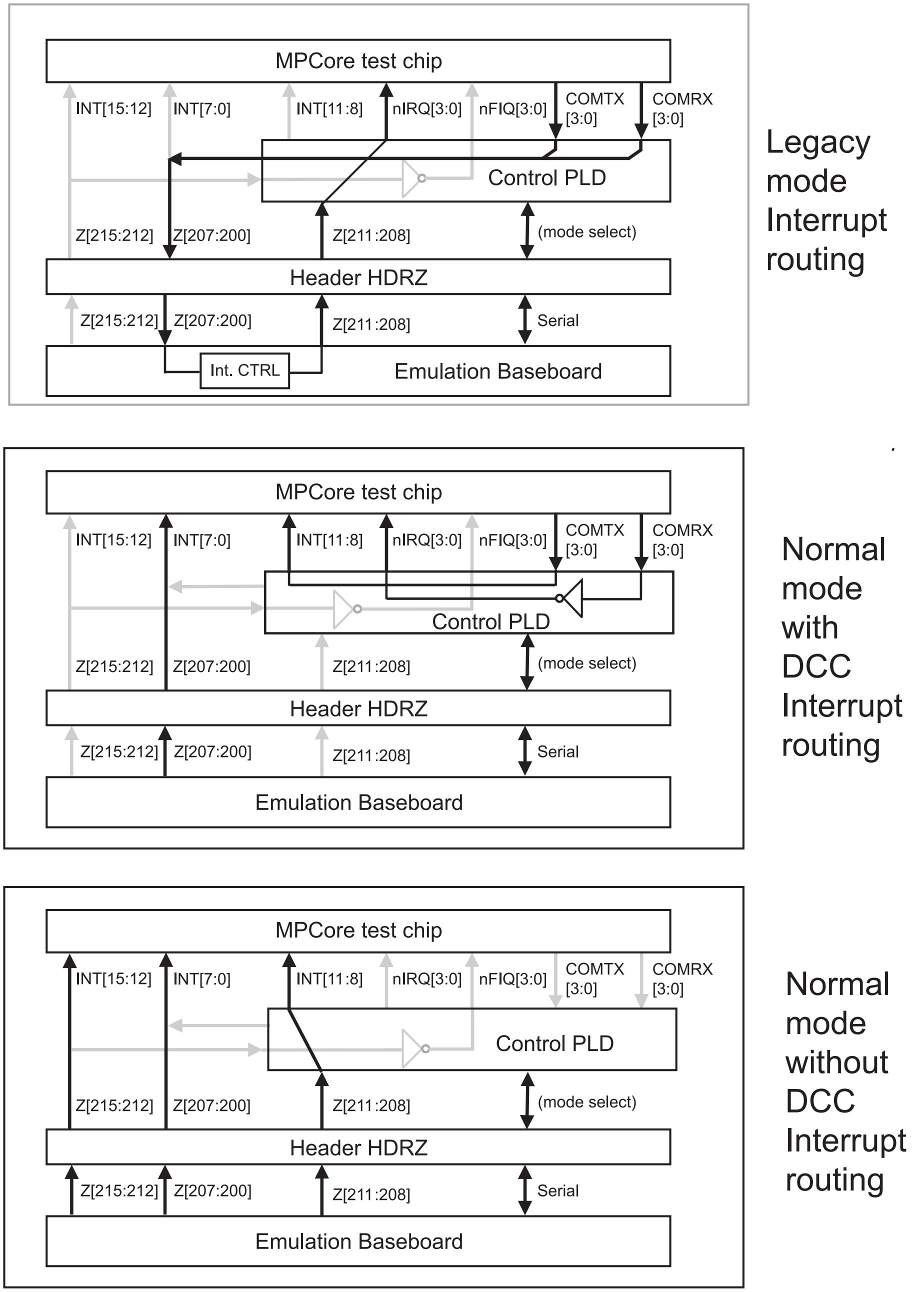


Figure 3-8 Interrupt mode signal routing

A total of 16 general purpose interrupt lines are provided at header HDRZ.

The CT11MPCore PLD routing functions are:

- to route the HDRZ header interrupt sources to the ARM11 MPCore test chip *distributed interrupt controller* inputs, **INT[15:0]**
  - — set the `INTMODE[2:0]` field in the SYS_PLD_CTL1 register to `bx00` to select *Legacy mode* interrupt routing
  - — set the `INTMODE[2:0]` field in the SYS_PLD_CTL1 register to `bx01` to select *Normal mode with DCC* interrupt routing
  - — set the `INTMODE[2:0]` field in the SYS_PLD_CTL1 register to `bx1x` to select *Normal mode without DCC* interrupt routing.

  Note

  INT[15:0] on the ARM11 MPCore test chip maps to **INT[47:32]** in the ARM11 MPCore *distributed interrupt controller*. See *ARM11 MPCore Processor Technical Reference Manual* (ARM DDI 0360) for further details.

- to route the CPU legacy IRQ request input lines, **nIRQ[3:0]**, for DCC interrupt handling

  Note

  The **COMMRX[3:0]** DCC interrupt sources are inverted by the CT11MPCore PLD.

- to route the Z[215:212] pins on header HDRZ to drive the CPU legacy fast interrupt request lines, **nFIQ[3:0]**
  - — set the `INTMODE[2:0]` field in the SYS_PLD_CTL1 register to `b1xx` to enable the **nFIQ[3:0]** interrupt lines.

  Note

  The CT11MPCore PLD inverts the Z[215:212] signals connected to **nFIQ[3:0]** when `INTMODE[2]` is HIGH. When `INTMODE[2]` is LOW, the CT11MPCore PLD disables the **nFIQ[3:0]** interrupt lines by forcing them HIGH.

Interrupt routing options

The interrupt routing options, between the ARM11 MPCore test chip and the EB, are shown in Table 3-2.

Table 3-2 Interrupt routing options

Interrupt	Legacy mode (bx00 default)	Normal mode with DCC (bx01)	Normal mode (bx1x)
INT 0	**COMMRX[0]**	**AACI**	**AACI**
INT 1	**COMMRX[1]**	**EB_TIMER0/1**	**EB_TIMER0/1**
INT 2	**COMMRX[2]**	**EB_TIMER2/3**	**EB_TIMER2/3**
INT 3	**COMMRX[3]**	**USB**	**USB**
INT 4	**COMMTX[0]**	**EB_UART0**	**EB_UART0**
INT 5	**COMMTX[1]**	**EB_UART1**	**EB_UART1**
INT 6	**COMMTX[2]**	**EB_RTC**	**EB_RTC**
INT 7	**COMMTX[3]**	**EB_KMI0** (KYBD)	**EB_KMI0** (KYBD)
INT 8	**EB_GIC1_nIRQ**	**COMMTX[0]**	**EB_KMI1** (MOUSE)
INT 9	**EB_GIC2_nIRQ**	**COMMTX[1]**	**EB_ETHINT**
INT 10	**USB**	**COMMTX[2]**	**EB_GIC1_nIRQ**
INT 11	**AACI**	**COMMTX[3]**	**EB_GIC2_nIRQ**
INT 12	**EB_GIC1_nFIQ**	**EB_GIC1_nFIQ**	**EB_GIC1_nFIQ**
INT 13	**EB_GIC2_nFIQ**	**EB_GIC2_nFIQ**	**EB_GIC2_nFIQ**
INT 14	**MMCI0**	**MMCI0**	**MMCI0**
INT 15	**MMCI1**	**MMCI1**	**MMCI1**
nIRQ[0]	**EB_GIC2_nIRQ**	**nCOMMRX[0]**	**1**
nIRQ[1]	**1**	**nCOMMRX[1]**	**1**
nIRQ[2]	**1**	**nCOMMRX[2]**	**1**

Table 3-2 Interrupt routing options (continued)

Interrupt	Legacy mode (bx00 default)	Normal mode with DCC (bx01)	Normal mode (bx1x)
nIRQ[3]	1	nCOMMRX[3]	1
EB_COMMRX	COMMRX[0]	0	0
EB_COMMTX	COMMTX[1]	0	0

3.5 HDRZ signal mux

The function of the HDRZ signal mux is to provide individual access to several of the ARM11 MPCore test chip signals, via header HDRZ, without introducing any significant system delays.

Note

Due to pin limitations, the EB system FPGA does not have direct access to these signals. As shown in Figure 3-9 on page 3-24, access is via a tile fitted at Tile Site 2 of the EB.

The structure of the mux and its connection to the EB is shown in Figure 3-9 on page 3-24. The positioning of the mux in the signal flow is shown in Figure 3-6 on page 3-14.

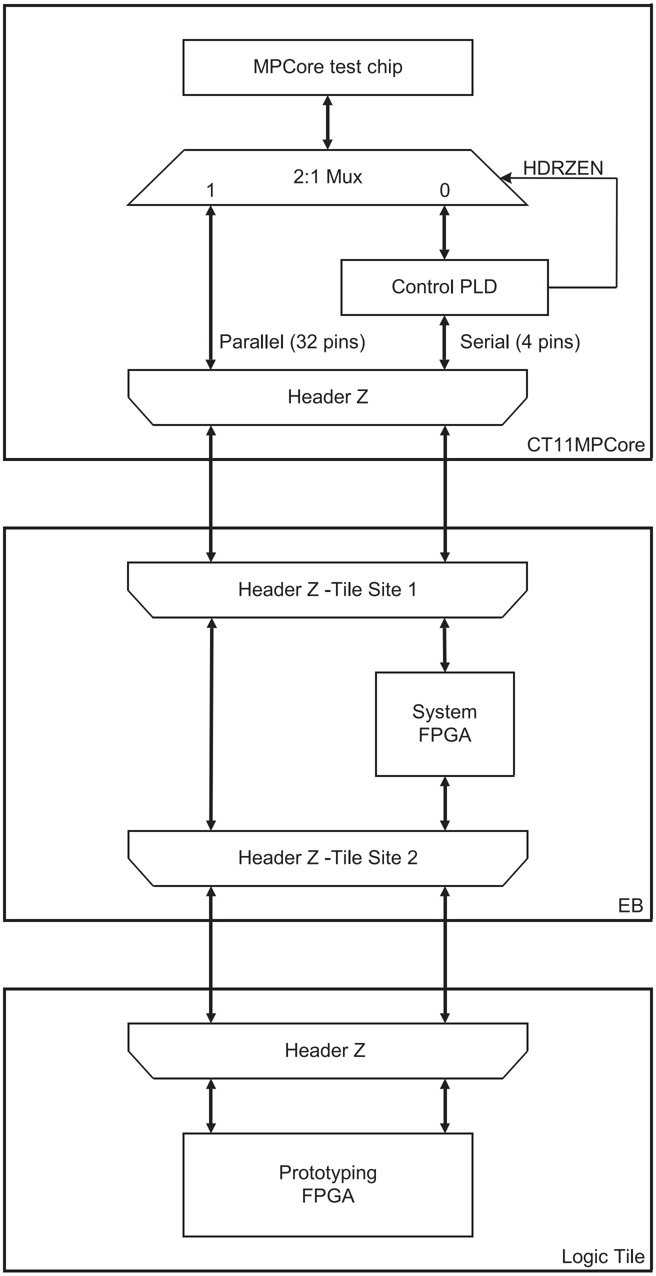


Figure 3-9 HDRZ signal mux

The HDRZ signal mux provides individual access to 32 ARM11 MPCore test chip signals. These signals are listed in Table 3-3.

Table 3-3 HDRZ signal mux connections

Z Bus	HDRZ pin	ARM11 MPCore test chip	Function
168	81	**nPORESET0**	Debug logic reset to ARM11 MPCore CPU 0
169	83	**nPORESET1**	Debug logic reset to ARM11 MPCore CPU 1
170	85	**nPORESET2**	Debug logic reset to ARM11 MPCore CPU 2
171	87	**nPORESET3**	Debug logic reset to ARM11 MPCore CPU 3
172	89	**nCPURESET0**	Reset to ARM11 MPCore CPU 0
173	91	**nCPURESET1**	Reset to ARM11 MPCore CPU 1
174	93	**nCPURESET2**	Reset to ARM11 MPCore CPU 2
175	95	**nCPURESET3**	Reset to ARM11 MPCore CPU 3
176	97	**RESETREQ0**	Watchdog reset request from ARM11 MPCore CPU 0
177	99	**RESETREQ1**	Watchdog reset request from ARM11 MPCore CPU 1
178	101	**RESETREQ2**	Watchdog reset request from ARM11 MPCore CPU 2
179	103	**RESETREQ3**	Watchdog reset request from ARM11 MPCore CPU 3
180	105	**nWDRESET0**	Watchdog reset to ARM11 MPCore CPU 0
181	107	**nWDRESET1**	Watchdog reset to ARM11 MPCore CPU 1
182	109	**nWDRESET2**	Watchdog reset to ARM11 MPCore CPU 2
183	111	**nWDRESET3**	Watchdog reset to ARM11 MPCore CPU 3

Table 3-3 HDRZ signal mux connections (continued)

Z Bus	HDRZ pin	ARM11 MPCore test chip	Function
184	113	**EDBGRQ0**	External debug request to ARM11 MPCore CPU 0
185	115	**EDBGRQ1**	External debug request to ARM11 MPCore CPU 1
186	117	**EDBGRQ2**	External debug request to ARM11 MPCore CPU 2
187	119	**EDBGRQ3**	External debug request to ARM11 MPCore CPU 3
188	121	**DBGACK0**	Debug acknowledge from ARM11 MPCore CPU 0
189	123	**DBGACK1**	Debug acknowledge from ARM11 MPCore CPU 1
190	125	**DBGACK2**	Debug acknowledge from ARM11 MPCore CPU 2
191	127	**DBGACK3**	Debug acknowledge from ARM11 MPCore CPU 3
192	128	**SMPnAMP0**	Indicates whether ARM11 MPCore CPU 0 is in AMP or SMP mode
193	126	**SMPnAMP1**	Indicates whether ARM11 MPCore CPU 1 is in AMP or SMP mode
194	124	**SMPnAMP2**	Indicates whether ARM11 MPCore CPU 2 is in AMP or SMP mode
195	122	**SMPnAMP3**	Indicates whether ARM11 MPCore CPU 3 is in AMP or SMP mode
196	120	**STANDBYWFI0**	Indicates whether ARM11 MPCore CPU 0 is in WFI state

Table 3-3 HDRZ signal mux connections (continued)

Z Bus	HDRZ pin	ARM11 MPCore test chip	Function
197	118	**STANDBYWFI1**	Indicates whether ARM11 MPCore CPU 1 is in WFI state
198	116	**STANDBYWFI2**	Indicates whether ARM11 MPCore CPU 2 is in WFI state
199	114	**STANDBYWFI3**	Indicates whether ARM11 MPCore CPU 3 is in WFI state

3.6 Power supply control

The EB system FPGA implements registers and a serial interface to the CT11MPCore PLD to enable you to:

- change the voltage supplied to the ARM11 MPCore test chip by writing values to the serially-programmed *digital to analog converters* (DACs)
- read CT11MPCore on-board voltages and currents from an 8-channel 12-bit *analog to digital converter* (ADC).

The voltage control and voltage and current monitoring scheme is shown in Figure 3-10.

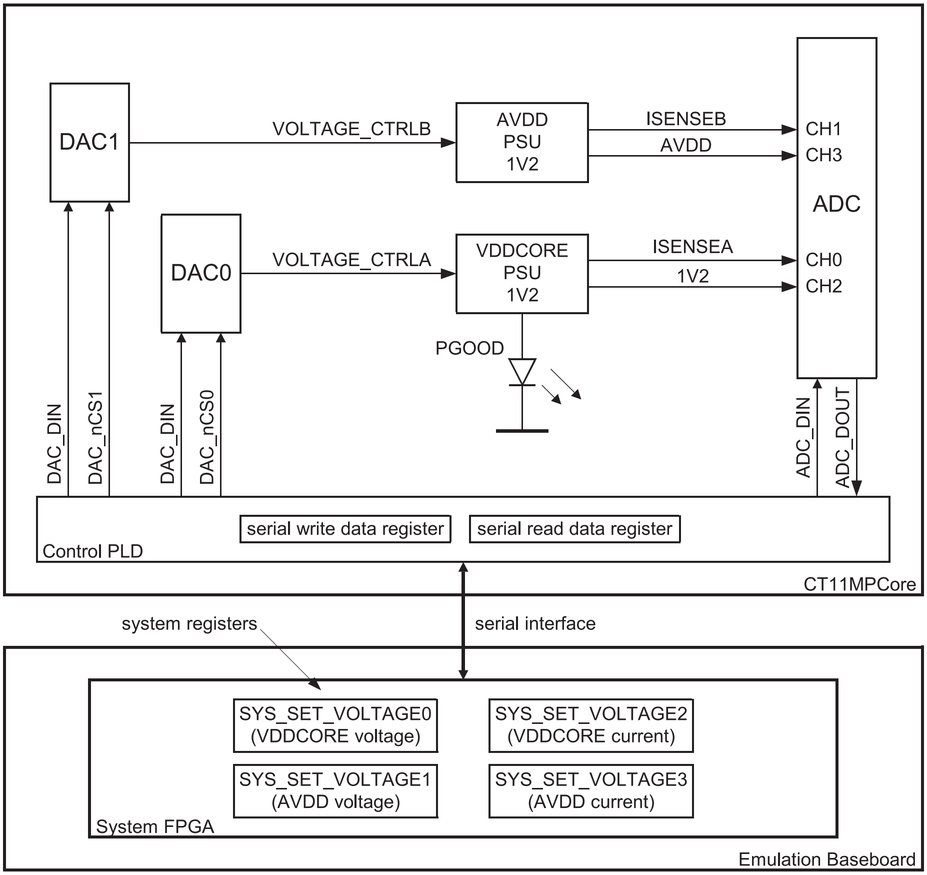


Figure 3-10 Voltage control and voltage and current monitoring

The serial write data register in the CT11MPCore PLD is updated continuously with values from the SYS_VOLTAGE registers in the EB system FPGA. These values control DAC0 and DAC1 that set the voltages applied to the ARM11 MPCore test chip.

The SYS_VOLTAGE registers in the EB system FPGA are continually updated with voltage and current monitoring values from the serial read data register in the CT11MPCore PLD.

The DAC and ADC data values are transferred via the 4-wire serial interface.

3.6.1 Setting the ARM11 MPCore test chip voltage

The core voltages depend on:

- The feedback resistors for the regulators
  - resistors R47, R49, and R83 are the feedback resistors for the **VDDCORE** 1V2 supply
  - resistors R54, R55, and R84 are the feedback resistors for the **AVDD** PLL 1V2 supply
  - resistors R62, R61 are the feedback resistors for the **1V8** supply.

  Note

  These resistors define the supply voltages at power-on and the adjustment range for each supply, These resistors are fitted at manufacture to give the correct core and PLL voltage for the ARM11 MPCore test chip.

- The values loaded into the PLD for the DAC settings. The values provide a positive or negative offset to the default power-on voltages.

  The output voltages are given by

  - **VDDCORE** `= 0.8V * ((1 + R47*[R49+R83]/[R49*R83]) - (Idac*R47))`
  - **AVDD** `= 0.6V * ((1 + R54*[R55+R84]/[R55*R84]) - (Idac*R54))`
  - **1V8** `= 0.6V * (1 + R62/R61)`

  Note

  `Idac` is the current sourced at **VOLTAGE_CTRLA** or **VOLTAGE_CTRLB** by the associated DAC. The current range is 0 to 50µA. At power up `Idac` is set to 25µA.

  The resistor values are typically chosen to give a ±0.25V adjustment range for the **VDDCORE** and **AVDD** PLL voltage. The default value loaded into the 8 bit DAC is `0x80`. A value of `0xFF` gives maximum negative offset from the default (-0.25V) and a value of `0x0` gives maximum positive offset from the default (+0.25V).

3.6.2 Reading the voltages and currents

The ADC on the CT11MPCore continuously reads the voltages and currents and updates the read serial register in the CT11MPCore PLD. The PLD sends these values over the 4-wire serial interface to the EB system FPGA.

The ADC is a 12 bit serial converter and uses an internal 2.5V reference. The LSB of the ADC reading corresponds to 610µV (the 2.5V reference of the ADC divided by 4096).

The formula to calculate the supply voltages ($V_{DDx)}$ is:

$$V_{DDx} = \mathrm{INT}(\mathrm{ADC}[11:0] * (610 * 10^{-6}))$$

Voltages proportional to the **VDDCORE** and **AVDD** currents are developed across the R_{sense} resistors R59 and R60 respectively. Each of the sense resistors has a voltage amplifier that has a fixed gain of 100. This is necessary because the voltage developed across the sense resistor is too low to measure directly with the ADC.

To calculate the current (I_{DDx}) drawn by a supply line use the formula:

$$I_{DDx} = V_{DDx} / R_{SENSE} * 100$$

Note

By default, the R_{SENSE} resistor is 0.025Ω for **VDDCORE** and 0.5Ω for **AVDD**. This provides a linear voltage to current relationship of 2.5V/Amp for **VDDCORE** and 50V/Amp for **AVDD**.

See *EB system FPGA registers* on page 3-46 for details of the **SYS_VOLTAGE2** and **SYS_VOLTAGE3** registers which monitor the **VDDCORE** and **AVDD** current sensing values.

PGOOD signal and LED

The power-good signal (**PGOOD**) from the **VDDCORE** supply step-down regulator is pulled LOW if the output voltage is not within 7.5%.

The **PGOOD** signal is buffered and drives the CT11MPCore on board PGOOD LED indicator, D1. See *LED indicators* on page 5-30 for details of all the CT11MPCore indicators.

3.7 AXI bus multiplexing

This section describes the AXI bus multiplexing. A multiplexing scheme is necessary to reduce the number of pins required on the HDRX and HDRY headers.

3.7.1 Multiplexing scheme

The CT11MPCore AXI bus multiplexing and demultiplexing scheme is shown in Figure 3-11.

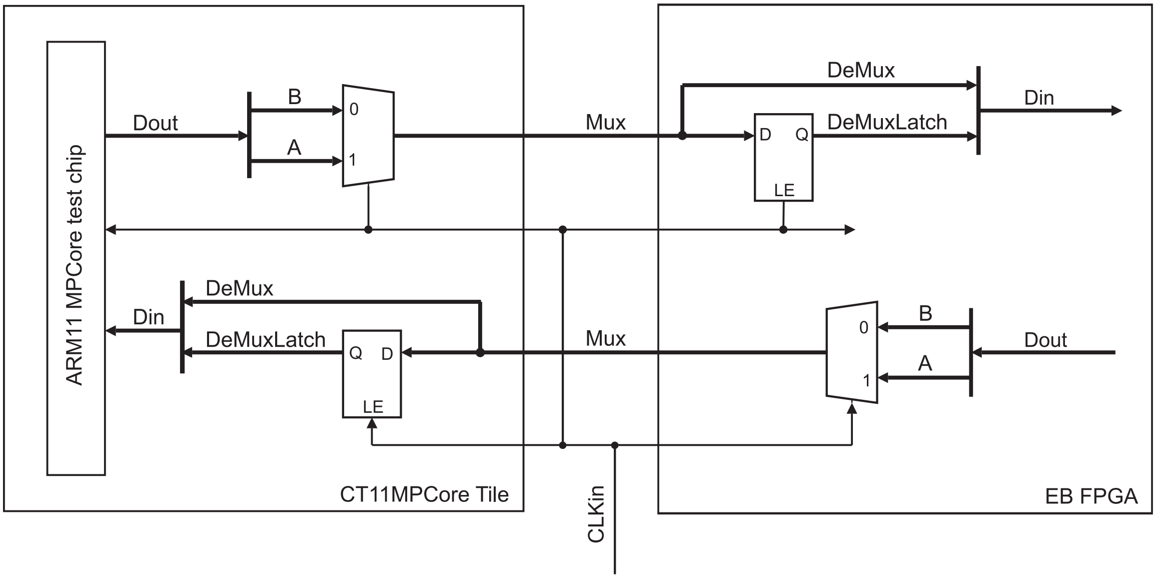


Figure 3-11 CT11MPCore AXI bus multiplexing scheme

Data is multiplexed as follows:

Dout is split bitwise into data **Dout/A** and **Dout/B** and is multiplexed onto **Mux** depending on the level of **CLKin**. When **CLKin** is HIGH, **Dout/A** is selected and when **CLKin** is LOW, **Dout/B** is selected. For example, when **CLKin** is HIGH, **RDATA0[31:0]** is selected and when **CLKin** is LOW, **RDATA0[63:32]** is selected. See *HDRY signals* on page 5-10 for details of the AXI port 0 multiplexed signals, and *HDRX signals* on page 5-3 for details of the AXI port 1 multiplexed signals.

Data is de-multiplexed as follows:

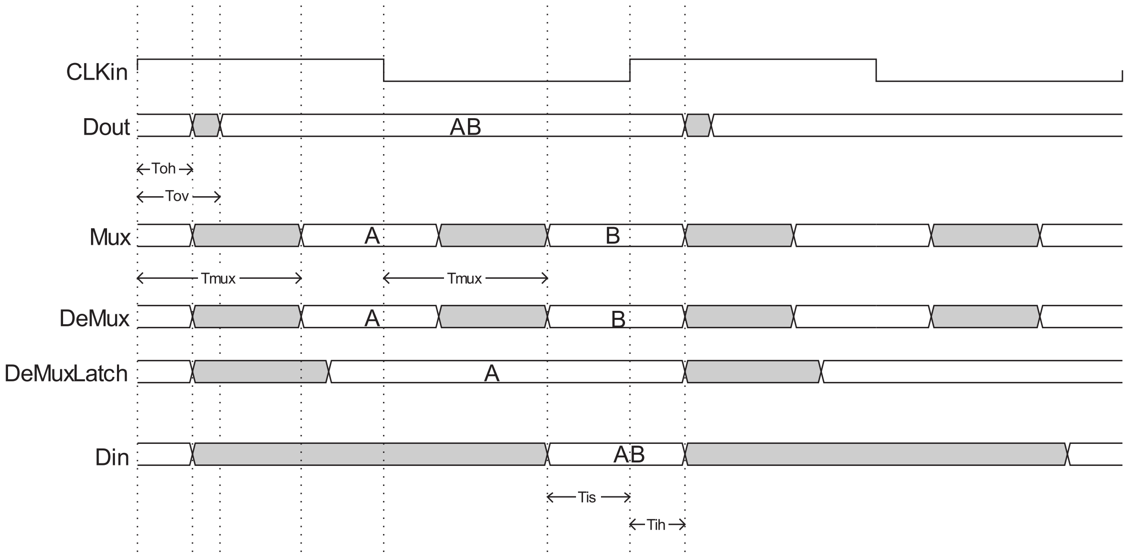
Dout/A is present on **Mux** when **CLKin** is HIGH and is latched onto **DeMuxLatch** when **CLKin** goes LOW. **Dout/B** is present on **Mux** when **CLKin** is LOW and is passed straight through as **DeMux**. **DeMuxLatch** and **DeMux** are combined bitwise as **Din**.

Note

CLKin is driven by distributed outputs from the ispClock5620 clock generators. The ispClock5620 clock generators are driven by **CLKOUTDIV** and **CLKOUTDIVD** from the ARM11 MPCore test chip.

This design requires that **Dout** is generated on the rising edge of **CLKin,** and that **Din** is captured on the rising edge of **CLKin**.

The multiplexer timing requirements at the board headers are shown in Figure 3-12 on page 3-33.



Timing requirements;

Toh	min = 0ns	(output hold)
Tov	max = 2ns	(output valid)
Tis	max = 2ns	(Inputsetup)
Tih	max = 0ns	(input hold)
Tmux	max = 6ns	(multiplexer and board delay)

The CLKin is the clock driven into the MPCore Test Chip from the Core Tile.
All I/O timing must be with respect to this clock.

Figure 3-12 CT11MPCore AXI mux timing

AXI multiplexing logic

The practical implementation of the AXI port 0 multiplexing is shown in Figure 3-13 on page 3-34.

Note

AXI port 0 from the ARM11 MPCore test chip can be isolated from HDRY upper and lower headers to make pins available for routing other signals between other tiles in the stack and the EB. Header isolation is achieved using the bus switches (PI3C34X245B) shown in Figure 3-13. These bus switches are not implemented in the AXI port 1 multiplexing logic.

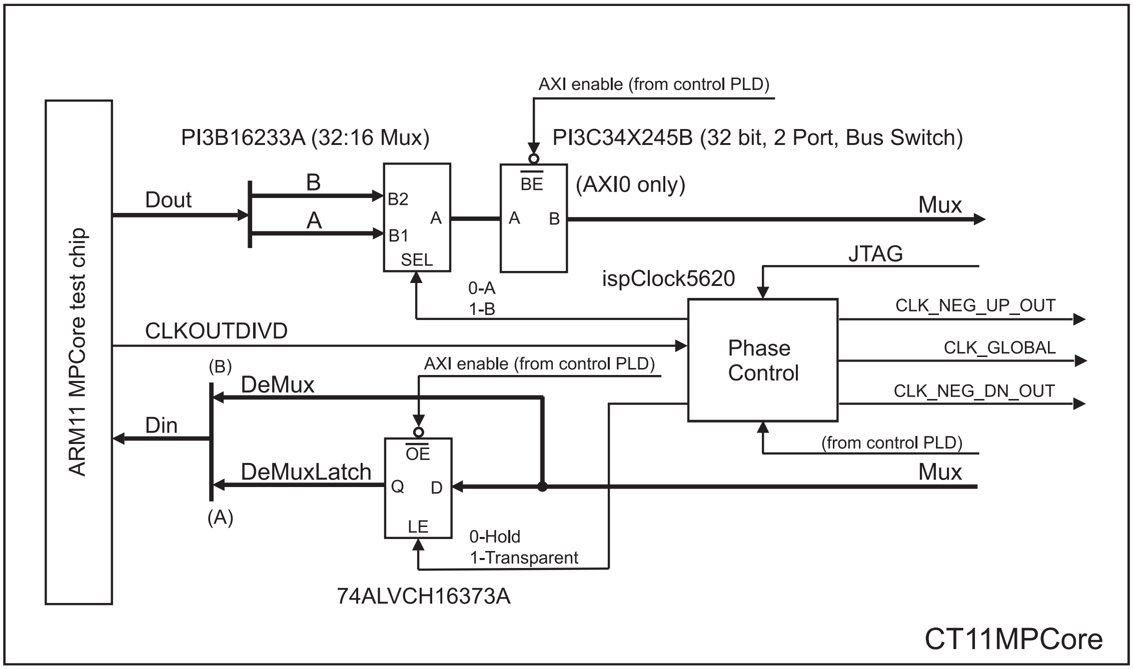


Figure 3-13 MPCore AXI multiplexing logic

The AXI port 0 bus clock, **CLKOUTDIVD** from the ARM11 MPCore test chip is distributed to the multiplexer, *PI3B16233A* and the latch, *74ALVCH16373A*. Phase control of these signals is critical for the correct operation of the mux/demux scheme at high system clock frequencies. Phase control is achieved with the ispClock5620 clock generator. This minimizes clock skew issues in the multiplexing logic. The clock generator non-volatile configuration settings are JTAG configurable when the CT11MPCore is in JTAG configuration mode. See *JTAG control and clock routing* on page 3-63 for further details.

Note

Two ispClock5620 clock generators are necessary to adequately distribute the AXI bus clock to all of the AXI bus multiplexers and latches. **CLKOUTDIV** from the ARM11 MPCore test chip is used as the AXI bus clock source for the second ispClock5620 clock generator. The second ispClock5620 clock generator also provides the external clocks. See *Clock routing* on page 3-4 for details.

AXI signal routing

The AXI signal routing scheme is as follows:

- the AXI Port 0 signals from the ARM11 MPCore test chip are 2:1 multiplexed onto pins Y0 through to Y143 on the CT11MPCore HDRY header
- the AXI Port 1 signals from the ARM11 MPCore test chip are 2:1 multiplexed onto pins X0 through to X143 on the CT11MPCore HDRX header.

See *HDRX signals* on page 5-3 and *HDRY signals* on page 5-10 for pinout information.

Note

When **AXInOE0** from the CT11MPCore PLD is set HIGH this isolates the multiplexed AXI Port 0 bus from the MPCore HDRY header allowing the Y bus to be used by other tiles. In this mode the CT11MPCore can use a single AXI bus (AXI port 1). Table 3-5 on page 3-39 gives details of the CT11MPCore PLD signals.

AWUSER and ARUSER sideband signal routing

These signals provide information about the L1 cache, write-back, and shared access.

The **AWUSER0[4:0]** and **ARUSER0[4:0]** sideband signals from AXI port 0 and the **AWUSER1[4:0]** and **ARUSER1[4:0]** sideband signals from AXI port 1 are 2:1 multiplexed onto pins on the CT11MPCore HDRZ header. The **AWUSER0[4:0]** and **AWUSER1[4:0]** buses are available when **CLKin** is HIGH and the **ARUSER0[4:0]** and **ARUSER1[4:0]** buses are available when **CLKin** is LOW. See *HDRZ signals* on page 5-16 for pinout information. Table 3-5 on page 3-39 gives details of the CT11MPCore PLD signals.

3.8 Overview of Core Tile configuration

The ARM11 MPCore test chip, clock source, voltage levels, and a number of system parameters are configurable on the CT11MPCore. In a final product, core configuration is static and the core configuration signals are tied HIGH or LOW and the voltage and clocks are fixed. However, the CT11MPCore allows you to program these signals for experimentation.

There are several ways that CT11MPCore configuration occurs:

- The CT11MPCore PLD.

  This is the primary configuration source. The PLD serial registers are described in *CT11MPCore serial registers* on page 3-46.

- CT11MPCore specific configuration registers in the EB system FPGA. These registers allow you to change several of the parameters controlled by the CT11MPCore PLD.

  These registers are described in *EB system FPGA registers* on page 3-46.

- Signals present on the Core Tile connector HDRZ.

  The HDRZ configuration signals are described in *HDRZ signal mux* on page 3-23

- Control registers in the ARM11 MPCore test chip.

  The control registers in the test chip are described in Chapter 4 *Test Chip Hardware Description*.

- Control registers in the ARM11 MPCore.

  For details of the registers in the ARM11 MPCore see the *ARM11 MPCore Processor Technical Reference Manual* (ARM DDI 0360).

3.8.1 CT11MPCore PLD signals

The CT11MPCore PLD performs the following functions:

- loading of data to the DACs that control the programmable power supplies (see *Power supply control* on page 3-28)
- reading of data from the ADCs that monitor the ARM11 MPCore test chip voltages (see *Reading the voltages and currents* on page 3-30)
- configuring the ARM11 MPCore test chip PLL on power up (see *Clocks* on page 3-4)
- controlling the individual resets to the MPCore processors (see *Resets* on page 3-11)

- controlling the interrupt mode that defines the interrupt routing (see *Interrupts* on page 3-18)
- configuring the L2, AXI and memory sub-systems within the ARM11 MPCore test chip (see Chapter 4 *Test Chip Hardware Description*)
- controlling the HDRZ signal multiplexers (see *HDRZ signal mux* on page 3-23).

Note

The binary image for the EB system FPGA depends on whether a tile is fitted in the remaining baseboard tile site. A further Core Tile or a Logic Tile may be fitted here for system prototyping. The application notes supplied include FPGA and PLD images for the current ARM supported tile combinations.

The CT11MPCore PLD is controlled by the serial interface signals listed in Table 3-4. These signals connect to the EB via the HDRZ header.

Table 3-4 PLD control signals

Signal	Description
PLDCLK	Clocks data into or out of the PLD
PLDD1	Serial data input to PLD
PLDD0	Serial data output from PLD
PLDnRESET	Resets the serial interface and signals the start of transfers

The EB system FPGA implements seven registers that hold values sent to and received from the CT11MPCore PLD using the 4-wire serial interface. The EB system FPGA provides the serialization and deserialization logic required for the interface. The interface timing is shown in Figure 3-14 on page 3-38.

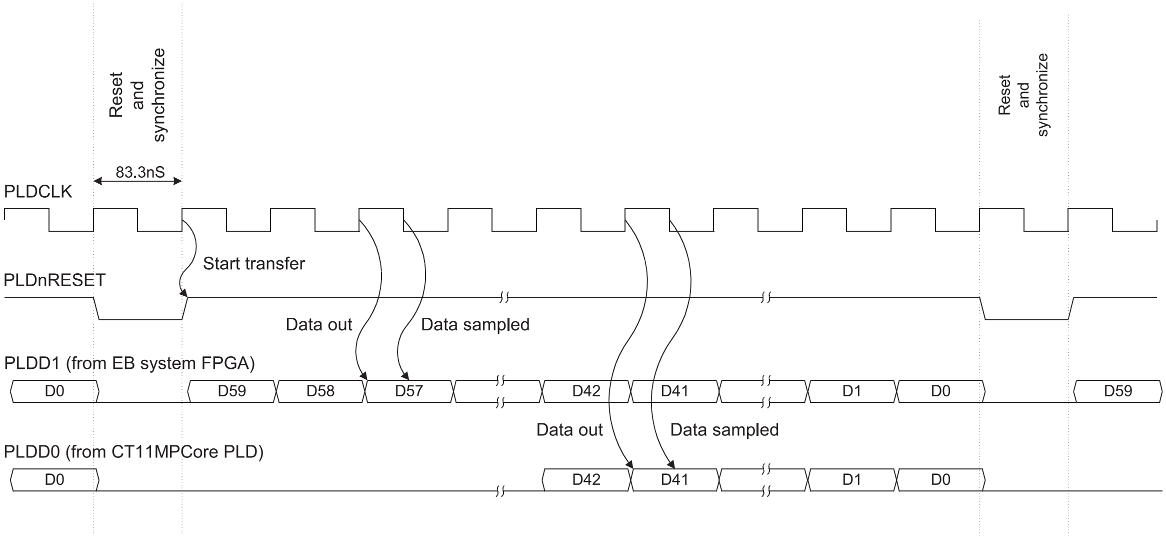


Figure 3-14 4-wire serial interface timing

Data is output on the rising edge of **PLDCLK** and sampled on the falling edge. The interface is reset and re-synchronized by **PLDnRESET** after each 60 bit serial transfer. The rising edge of **PLDnRESET** also indicates the start of a transfer. The data is transferred MSB first in both directions across the interface.

Note

The **PLDD0** serial data stream from the CT11MPCore PLD is only 43 bits long. Bits **PLDD0[42:0]** are used to transfer the data, **PLDD0[59:43]** are reserved and tied LOW.

The CT11MPCore PLD drives the configuration signals listed in Table 3-5 on page 3-39.

Note

There are a total of 60 bits used in the *Serial write data register* serial data stream.

Table 3-5 Serial write data register

PLD Pin Name	Serial Bits PLDD1	Definition
RD_DIV[3:0]	3:0	ARM11 MPCore test chip PLL Clock divider. See *Clock setting in reset mode* on page 3-6.
RD_CTRL[31:16]	19:4	ARM11 MPCore test chip PLL Clock control. See *Clock setting in reset mode* on page 3-6.
PLLUPDATE	20	Start PLL update flag (reserved for future PLL support).
L2BYPASS	21	Selects the L2 cache bypassing: • 0: you can use the L2 cache • 1: the L2 cache is bypassed.
L2MASTNUM	22	Selects the number of L220 master ports: • 0: one master port, M1. When one master is selected, master port 1 is used • 1: two master ports, M0 and M1 are used. The port options available are shown in Figure 4-11 on page 4-28. **Note** This bit is not currently user configurable and is set HIGH by the CT11MPCore PLD to select two L220 master ports when using the EB.
MP**MASTNUM**	23	Selects the number of ARM11 MPCore test chip master ports: • 0: one master port, M0. When one master is selected, master port 0 is used • 1: two master ports, M0 and M1 are used. The port options available are shown in Figure 4-11 on page 4-28.
AXInOE0	24	AXI Mux output enable: • 0: enable AXI Port 0 to Y headers • 1: disable and isolate AXI Port 0 from Y headers. See *AXI bus multiplexing* on page 3-31.
HDRZEN	25	Header Z optional signal mux enable: • 0: disable optional HDRZ signals • 1: enable optional HDRZ signals. See *HDRZ signal mux* on page 3-23.

Table 3-5 Serial write data register (continued)

PLD Pin Name	Serial Bits PLDD1	Definition
VINITHI[3:0]	29:26	When HIGH, indicates high-Vecs mode for the particular ARM11 MPCore processor.
CFGEND[1:0]	31:30	The U and EE bits reset values in the CP15 Control Register and E Bit reset value in CPSR/SPSR depend on the value of CFGEND[1:0]. See the *ARM11 MPCore Processor Technical Reference Manual* (ARM DDI 0360) for details.
nCPURESET[3:0]	35:32	Individual ARM11 MPCore processor resets.
DBGMUX[11:0]	47:36	Debug matrix select. Selects the external debug request and debug acknowledge routing between the four ARM11 MPCore CPUs to enable cross-triggering.
INTMODE[2:0]	50:48	Interrupt mode: • bx00: Legacy • bx01: New + DCC • bx1x: New no DCC • b1xx: FIQ[3:0] enable. See *Interrupts* on page 3-18.
DACDAT[7:0]	58:51	Voltage control, 8 bit DAC value applied to the DAC channel selected by **DACSEL**. See *Setting the ARM11 MPCore test chip voltage* on page 3-29.
DACSEL	59	Selects the voltage control DAC channel. See *Setting the ARM11 MPCore test chip voltage* on page 3-29.

The CT11MPCore PLD *Serial read data register* provides the status signals listed in Table 3-6 on page 3-41.

Note

There are a total of 43 bits used in the *Serial read data register* data stream.

Table 3-6 Read serial data register

PLD Pin Name	Serial Bits PLDD0	Definition
PLOCK	0	ARM11 MPCore test chip PLL lock indicator. • 0: ARM11 MPCore test chip PLL is not locked to **REFCLK** • 1: ARM11 MPCore test chip PLL is locked to **REFCLK**. See *Clock setting in reset mode* on page 3-6.
ISP0nLOCK	1	ispClock5620 PLL lock indicator 0. • 0: ispClock5620 PLL 0 is locked to **CLKOUTDIVD** • 1: ispClock5620 PLL 0 is not locked to **CLKOUTDIVD**. See *Clocks* on page 3-4.
ISP1nLOCK	2	ispClock5620 PLL lock indicator 1. • 0: ispClock5620 PLL 1 is locked to **CLKOUTDIV** • 1: ispClock5620 PLL 1 is not locked to **CLKOUTDIV**. See *Clocks* on page 3-4.
PLLUPDATED	3	MPCore PLL settings status. • 0: ARM11 MPCore test chip PLL settings are being updated. • 1: ARM11 MPCore test chip PLL settings are complete.
RESETREQ[3:0]	7:4	Individual watchdog reset requests, one from each ARM11 MPCore CPU
SMPnAMP[3:0]	11:8	Individual AMP or SMP mode indicators, one from each MPCore CPU. • 0: Indicates AMP mode (processor is not part of coherency) • 1: Indicates SMP mode (processor is part of coherency) See the *ARM11 MPCore Processor Technical Reference Manual* (ARM DDI 0360).
STANDBYWFI[3:0]	15:12	Individual WFI indicators. Indicates if an MPCore CPU is in WFI state. See the *ARM MPCore Processor Technical Reference Manual* (ARM DDI 0360).
COMMTX[3:0]	19:16	Comms channel transmit from each ARM11 MPCore CPU.
COMMRX[3:0]	23:20	Comms channel receive from each ARM11 MPCore CPU.
ADCDAT[11:0]	35:24	Current sensing 12-bit ADC value (selected by **ADCSEL[2:0]**).

Table 3-6 Read serial data register (continued)

PLD Pin Name	Serial Bits PLDD0	Definition
ADCSEL[2:0]	38:36	ADC channel currently being converted (**ADCDAT** value is from ADC channel **ADCSEL - 1**).
PLDVER[3:0]	42:39	PLD build version.
Reserved	59:43	Legacy only (not used).

3.8.2 Core configuration from ARM CP15

The control coprocessor, CP15, implements a range of control functions and provides status information for the MPCore Multiprocessor.

The main functions controlled by CP15 are:

- overall system control and configuration of the MPCore
- cache configuration and management
- Memory Management Unit (MMU) configuration and management
- system performance monitoring.

See the *ARM11 MPCore Processor Technical Reference Manual* (ARM DDI 0360) for details on the CP15 registers.

3.9 Memory configuration

The memory subsystems implemented by the ARM11 MPCore test chip are:

MPCore Level 1 (L1)

The MPCore memory subsystem has 32KB of L1 instruction cache and 32KB of L1 data cache per CPU.

L220 cache controller Level 2 (L2)

The L220 cache controller memory subsystem has 1MB of L2 unified cache.

There is no provision for PISMO Expansion Memory Modules on the CT11MPCore, System *Level 3* (L3) memory is located on the EB. If additional L3 memory is required, you can add PISMO Expansion Memory Modules to the EB. Additional L3 memory may also be available via the EB second tile site, see *Realview Emulation Baseboard User Guide* (ARM DUI 0303) for system memory details.

For details of the L1 and L2 memory subsystems implemented in the ARM11 MPCore test chip see *Memory configuration* on page 4-23.

For general information about the L1 and L2 memory subsystems see the *ARM11 MPCore Processor Technical Reference Manual* (ARM DDI 0360) and the *L220 Cache Controller Technical Reference Manual* (ARM DDI 0329).

3.10 Register configuration

This section describes the CT11MPCore specific registers and contains the following subsections:

- *ARM11 MPCore test chip registers*
- *CT11MPCore serial registers* on page 3-46
- *EB system FPGA registers* on page 3-46

3.10.1 ARM11 MPCore test chip registers

The ARM11 MPCore test chip peripheral space consists of four reserved pages of 4KB.

PERIPHBASE[17:0] is an input to the test chip that defines the base address of the 16KB of peripheral memory space. Table 3-7 on page 3-45 lists all the available peripheral accesses.

The register base address **PERIPHBASE [17:0]** is configurable via links on the CT11MPCore and defines bits [31:14] of the full 32 bit peripheral address.

The default peripheral base address setting is:

```
Address=0x1F000000
```

Note

To reduce the number of pins, the ARM11 MPCore test chip integrates some AXI decoding logic for selecting the Level 2 Cache Controller (L2CC) peripheral port and the ARM11 MPCore test chip specific registers. See *Peripheral decoder* on page 4-29 for details.

Table 3-7 Peripheral accesses

Address range	Peripheral accessed

[PERIPHBASE, PERIPHBASE + 8KB -1]
`Default range:`
`0x1F000000 - 0x1F001FFF`
	Snoop Control Unit (SCU) control register space for MPCORE:
	Address[31:14]=PERIPHBASE[17:0]
	Address[13]=0
	See the *ARM11 MPCore Processor Technical Reference Manual* for register details and address offset values.

[PERIPHBASE + 8KB, PERIPHBASE + 12KB -1]
`Default Range:`
`0x1F002000 - 0x1F002FFF`
	L220 control register space:
	Address[31:14]=PERIPHBASE[17:0]
	Address[13:12]=b10
	See the *L220 Cache Controller Technical Reference Manual* for register details and address offset values.

[PERIPHBASE + 12KB, PERIPHBASE + 16KB -1]
`Default Range:`
`0x1F003000 - 0x1F003FFF`
	Test chip specific registers:
	Address[31:14]=PERIPHBASE[17:0]
	Address[13:12]=b11
	Address[11:0] is as follows:
	`0x000`: Test chip PLL control register. See Table 4-1 on page 4-9.
	`0x004`: Test chip interrupt control register. See Table 4-6 on page 4-18.
	`0x008`: Test chip cluster ID register.
	`0x00C`: Test chip power status register. See Table 4-9 on page 4-21.
	`0x010`: Test chip way map register (not use on CT11MPCore)
	`0x014`: Test chip clock divider register. See Table 4-2 on page 4-12.
	Note
	The ARM11 MPCore test chip cluster ID register is provided for systems that use more than one ARM11 MPCore. Bits [3:0] of this register drive the ARM11 MPCore **CLUSTERID[3:0]** inputs. The default single ARM11 MPCore cluster ID is `0x0`. See the *ARM11 MPCore Processor Technical Reference Manual for more information*.

Caution

When you access the ARM11 MPCore test chip peripheral space, you must use single accesses of 32-bit aligned type, otherwise an abort, due to slave error, is generated.

3.10.2 CT11MPCore serial registers

The CT11MPCore PLD implements two registers, the *Serial write data register* and the *Serial read data register*. These registers mirror the content of CT11MPCore specific registers implemented in the EB system FPGA. The registers are continually updated via a 4-wire serial link to ensure that the content of the CT11MPCore and EB registers remains coherent.

The bit allocations for the *Serial write data register* are shown in *Serial write data register* on page 3-39. The bit allocations for the *Serial read data register* are shown in *Read serial data register* on page 3-41.

Note

The *Serial write data register* and the *Serial read data register* are only accessible via the 4-wire serial link. Changing or reading the contents of these registers is only possible by accessing the registers in the EB system FPGA. Not all fields in the *Serial write data register* are user configurable, some are pre-fixed by the EB system FPGA.

3.10.3 EB system FPGA registers

The CT11MPCore specific registers implemented in the EB system FPGA are:

SYS_PLD_CTRL1

This register sets the PLD serial write data register fields

- **HDRZEN**
- **MPMASTNUM**
- **L2BYPASS**
- **INTMODE[2:0]**
- **DBGMUX[11:0]**
- **nCPURESET[3:0]**
- **CFGEND[1:0]**
- **VINITHI[3:0]**

SYS_PLD_CTRL2

This register reads the PLD serial read data register fields

- **COMMRX[3:0]**
- **COMMTX[3:0]**
- **STANDBYWFI[3:0]**
- **SMPnAMP[3:0]**

- **RESETREQ[3:0]**
- **PLDVER[3:0]**

SYS_PLD_INIT

This register defines the ARM11 MPCore test chip *Test chip PLL control register* and *Test chip clock divider register* hardware initialization values.

The *Test chip PLL control register* initialization fields are:
PB[3:0]
PA[3:0]
N[3:0]
M[3:0]
PBSTBY
STBY
PLLEN

The *Test chip clock divider register* initialization fields are:
CLKOUTDIV[3:0]

Note

The SYS_PLD_INIT register only loads the *Test chip PLL control register* and *Test chip clock divider register* after the EB nPB reset button is pressed.

The ARM11 MPCore test chip registers can also be accessed directly to control the PLL and clock divider once out of reset. See *Clocks* on page 4-4 for details on accessing the ARM11 MPCore test chip clock control registers.

SYS_VOLTAGE0

This register reads and sets the **VDDCORE** voltage fields

- **ADC_DATA[11:0]**
- **DAC_DATA[7:0]**

SYS_VOLTAGE1

This register reads and sets the **AVDD** (PLL) voltage fields

- **ADC_DATB[11:0]**
- **DAC_DATB[7:0]**

SYS_VOLTAGE2

This register reads the **VDDCORE** current field

- **ADC_DATC[11:0]**
- **DAC_DATC{7:0]** (not used)

SYS_VOLTAGE3

This register reads the **AVDD** (PLL) current field

- **ADC_DATD[11:0]**
- **DAC_DATD[7:0]** (not used)

The EB SYS register base address, SYSBASE, is `0x10000000`.

The EB system FPGA registers are listed in Table 3-8.

Table 3-8 EB system FPGA registers

Name	Address	Description
SYS_PLD_CTRL1	`0x10000074`	This register sets the PLD serial write data register fields. See *SYS_PLD_CTRL1* on page 3-49 for details.
SYS_PLD_CTRL2	`0x10000078`	This register reads the PLD serial read data register fields. See *SYS_PLD_CTRL2* on page 3-51 for details.
SYS_PLD_INIT	`0x1000007C`	This register sets the ARM11MPCore test chip *Test chip PLL control register* and *Test chip clock divider register* hardware initialization values. See *SYS_PLD_INIT* on page 3-52 for details.
SYS_VOLTAGE0	`0x100000A0`	This register reads and sets the **VDDCORE** voltage fields. See *SYS_VOLTAGE0* on page 3-56 for details.
SYS_VOLTAGE1	`0x100000A4`	This register reads and sets the **AVDD** (PLL) voltage fields. See *SYS_VOLTAGE1* on page 3-57 for details.
SYS_VOLTAGE2	`0x100000A8`	This register reads the **VDDCORE** current field. See *SYS_VOLTAGE2* on page 3-58 for details.
SYS_VOLTAGE3	`0x100000AC`	This register reads the **AVDD** (PLL) current field. See *SYS_VOLTAGE3* on page 3-58 for details.

Note

All writable EB SYS registers must be unlocked first.
Write `0xA05F` to the SYS_LOCK register located at SYSBASE+`0x20` as follows:

Address [31:0]=0x10000020, Data [15:0]=0xA05F

Once unlocked, the registers will remain unlocked until the EB is reset.

SYS_PLD_CTRL1

This register is located at SYSBASE+`0x74` as follows:

`Address [31:0]=0x10000074`

Note

This register is locked for write by SYS_LOCK.
Write `0xA05F` to `0x10000020` to unlock until next EB reset.

Figure 3-15 shows the SYS_PLD_CTRL1 register bit assignments.

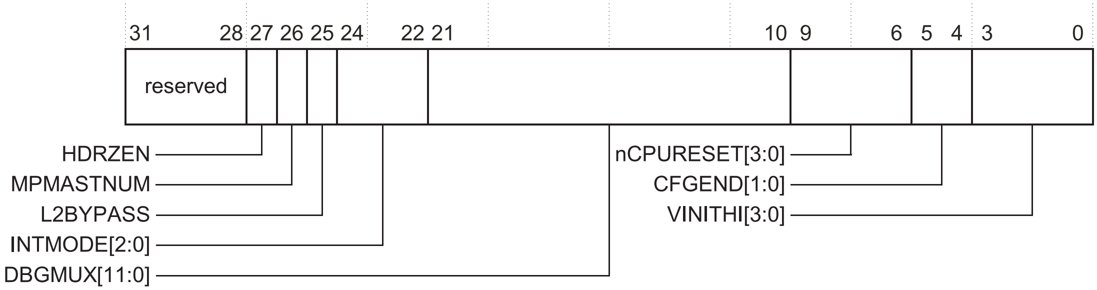


Figure 3-15 SYS_PLD_CTRL1 register

Table 3-9 lists the SYS_PLD_CTRL1 register bit assignments.

Table 3-9 SYS_PLD_CTRL1 register

Bits	Access	Name	Reset value	Description
[31:28]	Write ignored	–	`0x0`	reserved.
[27]	Write only Lockable	HDRZEN	`b0`	Header Z optional signal mux enable: • 0: disable optional HDRZ signals • 1: enable optional HDRZ signals. See *HDRZ signal mux* on page 3-23.
[26]	Write only Lockable	MPMASTNUM	`b0`	Selects the number of ARM11 MPCore test chip master ports: • 0: one master port, M0. When one master is selected, master port 0 is used • 1: two master ports, M0 and M1 are used. See *L220 bypass module* on page 4-26.
[25]	Write only Lockable	L2BYPASS	`b0`	Selects the L2 cache bypassing: • 0: you can use the L2 cache • 1: the L2 cache is bypassed. See *L220 bypass module* on page 4-26.
[24:22]	Write only Lockable	INTMODE	`b000`	Interrupt mode: • bx00: Legacy • bx01: New with DCC • bx1x: New no DCC • b1xx: FIQ[3:0] enable (independent of INTMODE[1:0]). See *Interrupts* on page 3-18.
[21:10]	Write only Lockable	DBGMUX	`0x000`	Sets the user defined debug cross-trigger mode. The required ARM11 MPCore **EDBGRQ[3:0]** and **DBGACK[3:0]** signals are routed using a debug matrix in the CT11MPCore PLD. See *Debug cross-trigger matrix* on page 4-20.

Table 3-9 SYS_PLD_CTRL1 register (continued)

Bits	Access	Name	Reset value	Description
[9:6]	Write only Lockable	nCPURESET	b1111	Individual MPCore CPU resets. nCPURESET[x] resets CPU[x}. See *Resets* on page 3-11.
[5:4]	Write only Lockable	CFGEND	b00	The U and EE bits reset values in the CP15 Control Register and E Bit reset value in CPSR/SPSR depend on the value of CFGEND[1:0]. See the *ARM11 MPCore Processor Technical Reference Manual* (ARM DDI 0360).
[3:0]	Write only Lockable	VINITHI	b0000	When HIGH, indicates high-Vecs mode for the respective MPCore CPU. See *ARM11 MPCore Processor Technical Reference Manual* (ARM DDI 0360).

SYS_PLD_CTRL2

This register is located at SYSBASE+`0x78` as follows:

```
Address [31:0]=0x10000078
```

Figure 3-16 shows the SYS_PLD_CTRL2 register bit assignments.

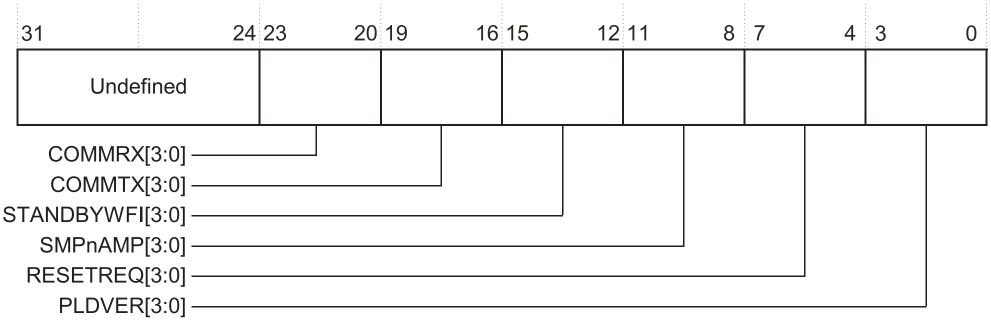


Figure 3-16 SYS_PLD_CTL2 register

Table 3-10 lists the SYS_PLD_CTRL2 register bit assignments.

Table 3-10 SYS_PLD_CTRL2 register

Bits	Access	Name	Reset value	Description
[31:24]	Read as zero, write ignored.	-	`0x00`	Undefined.
[23:20]	Read Only	COMMRX	bxxxx	Comms channels receive.
[19:16]	Read Only	COMMTX	bxxxx	Comms channels transmit.
[15:12]	Read Only	STANDBYWFI	bxxxx	Individual WFI indicators. Indicates if an ARM11 MPCore CPU is in WFI state. See the *ARM11 MPCore Processor Technical Reference Manual* (ARM DDI 0360).
[11:8]	Read Only	SMPnAMP	bxxxx	Individual AMP or SMP mode indicators, one from each ARM11 MPCore CPU. • 0: Indicates AMP mode (processor is not part of coherency) • 1: Indicates SMP mode (processor is part of coherency) See the *ARM11 MPCore Processor Technical Reference Manual* (ARM DDI 0360).
[7:4]	Read Only	RESETREQ	bxxxx	Individual watchdog reset requests, one from each ARM11 MPCore CPU.
[3:0]	Read Only	PLDVER	bxxxx	PLD build version.

SYS_PLD_INIT

This register is located at SYSBASE+`0x7C` as follows:

```
Address [31:0]=0x1000007C
```

Note

This register is locked for write by SYS_LOCK.
Write `0xA05F` to `0x10000020` to unlock until next EB reset.

SYS_PLD_INIT values will only be updated after nPB reset.

Figure 3-17 on page 3-53 shows the SYS_PLD_INIT register bit assignments.

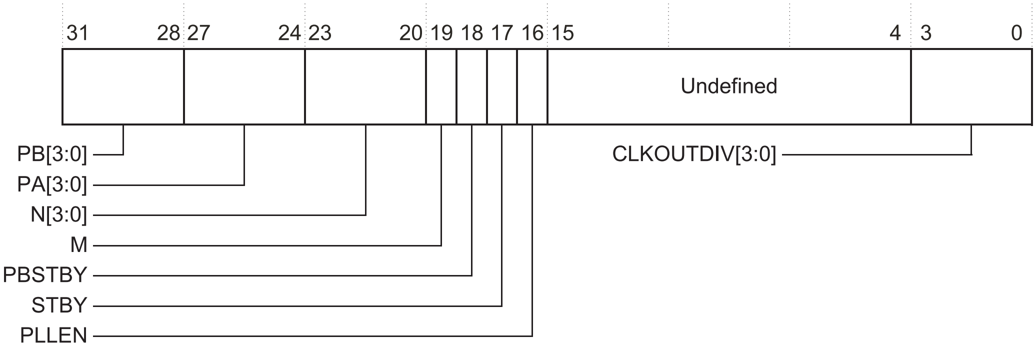


Figure 3-17 SYS_PLD_INIT register

Table 3-11 lists the SYS_PLD_INIT register bit assignments.

Table 3-11 SYS_PLD_INIT register

Bits	Access	Name	Reset value	Description
[31:28]	Read/Write Lockable	PB divider	`b0000`	PB divider ratio that the PLL uses: 0000: pb-value is 1 0001: pb-value is 2 … 1111: pb-value is 16.
[27:24]	Read/Write Lockable	PA divider	`b0000`	PA divider ratio that the PLL uses: 0000: pa-value is 1 0001: pa-value is 2 … 1111: pa-value is 16.
[23:20]	Read/Write Lockable	N divider	`b1011`	N divider ratio that the PLL uses: 0000: n-value is 1 0001: n-value is 2 … 1111: n-value is 16.
[19]	Read/Write Lockable	M divider	`b0`	M divider ratio that the PLL uses: • 0: m-value is 1 • 1: m-value is 2.

Table 3-11 SYS_PLD_INIT register (continued)

Bits	Access	Name	Reset value	Description
[18]	Read/Write Lockable	PBSTBY	`b0`	- 0: **CLKB** from PLL is active - 1: **CLKB** from PLL is 0.
[17]	Read/Write Lockable	STBY	`b0`	- 0: **CLKA** from PLL is active - 1: **CLKA** from PLL is 0.
[16]	Read/Write Lockable	PLLEN	`b1`	- 0: if *committed* and bit 0 of the PLL control test data register is 0 (its reset value) **REFCLK** will be selected as **CLKOUT** on the next rising edge of **REFCLK** - 1: if *committed*, and bit 0 of the PLL control test data register is 0 (its reset value) the PLL will be selected as **CLKOUT** on the next rising edge of **REFCLK**. PLL enable is *committed* when **STANDBYWFI[3:0]** from the `mpcore_tc_module` == b1111 or the input signal **CONFIGINIT** == b1.
[15:4]	Read as zero, write ignored.	-	`0x000`	Undefined
[3:0]	Read/Write Lockable	CLKOUTDIV	`b0000`	The clock divider ratio to drive the AXI bus clock and the FPGA: 0000: **CLKOUTDIV** is equal to **CLKIN** 0001: **CLKOUTDIV** is **CLKIN** divided by 2 0010: **CLKOUTDIV** is **CLKIN** divided by 3 0011: **CLKOUTDIV** is **CLKIN** divided by 4 0100: **CLKOUTDIV** is **CLKIN** divided by 5 0101: **CLKOUTDIV** is **CLKIN** divided by 6 0110: **CLKOUTDIV** is **CLKIN** divided by 7 0111: **CLKOUTDIV** is **CLKIN** divided by 8 1000: **CLKOUTDIV** is **CLKIN** divided by 1 1001: **CLKOUTDIV** is **CLKIN** divided by 10 1010: **CLKOUTDIV** is **CLKIN** divided by 3 1011: **CLKOUTDIV** is **CLKIN** divided by 12 1100: **CLKOUTDIV** is **CLKIN** divided by 5 1101: **CLKOUTDIV** is **CLKIN** divided by 14 1110: **CLKOUTDIV** is **CLKIN** divided by 7 1111: **CLKOUTDIV** is **CLKIN** divided by 16

Changing the core to AXI bus clock ratio

The MPCore test chip PLL register value may be set by software to change the core to AXI bus clock ratio. The reference clock from the EB, **REFCLK** is a 20MHz clock. In Table 3-12, the PLL feedback divider and the test chip output divider values are always equal. For example, if the AXI bus runs at 20MHz and the core at 240MHz, this requires both dividers to divide by 12. The required SYS_PLD_INIT value is ratio option 12:1 in Table 3-12. The SYS_PLD_INIT register is described in *EB system FPGA registers* on page 3-46.

You can change the core to AXI clock frequency ratio to match a real system design as follows:

1. Write `0xA05F` to `0x10000020` to unlock the SYS_PLD_INIT register.
2. Write the new value to the SYS_PLD_INIT register at `0x1000007C` – example PLL values are given in Table 3-12.
3. Shut down the debugger and press the RESET push button on the EB to load the new setting.

Table 3-12 SYS_PLD_INT register values

Ratio option (core to bus)	SYS_PLD_INIT value
1:1	`0xFF010000`
2:1	`0x77110001`
3:1	`0x55210002`
4:1	`0x33310003`
5:1	`0x22410004`
6:1	`0x22510005`
7:1	`0x22610006`
8:1	`0x11710007`
10:1	`0x11910009`
12:1	`0x11B1000B`
14:1	`0x00D1000D`
16:1	`0x00F1000F`

SYS_VOLTAGE0

This register is located at SYSBASE+`0xA0` as follows:

```
Address [31:0]=0x100000A0
```

Note

This register is locked for write by SYS_LOCK.
Write `0xA05F` to `0x10000020` to unlock until next EB reset.

Figure 3-18 shows the SYS_VOLTAGE0 register bit assignments.

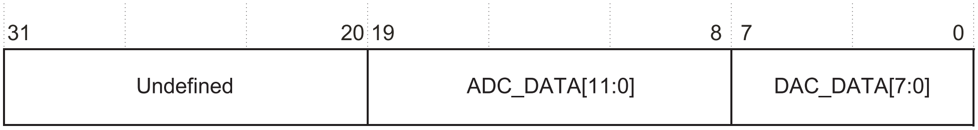


Figure 3-18 SYS_VOLTAGE0 register

Table 3-13 lists the SYS_VOLTAGE0 register bit assignments.

Table 3-13 SYS_VOLTAGE0 register

Bits	Access	Name	Reset value	Description
[31:20]	Read as zero, write ignored.	-	`0x000`	Undefined
[19:8]	Read only	ADC_DATA	`0x000`	Reads the **VDDCORE** voltage for the MPCore. The LSB of the ADC reading corresponds to 610µV. The formula to calculate the supply voltage (`VDDCORE`) is: `VDDCORE=INT(ADC_DATA[11:0]*(610*`10^{-6}`))`
[7:0]	Read/Write Lockable	DAC_DATA	`0x80`	Sets the **VDDCORE** voltage for the MPCore. The default value is 1.2V. Circuit values are typically chosen to give a ±0.25V adjustment range for the core voltage. A value of `0xFF` gives maximum negative offset from the default (-0.25V) and a value of `0x0` gives maximum positive offset from the default (+0.25V).

SYS_VOLTAGE1

This register is located at SYSBASE+0xA4 as follows:

```
Address [31:0]=0x100000A4
```

Note

This register is locked for write by SYS_LOCK.
Write 0xA05F to 0x10000020 to unlock until next EB reset.

Figure 3-19 shows the SYS_VOLTAGE1 register bit assignments.

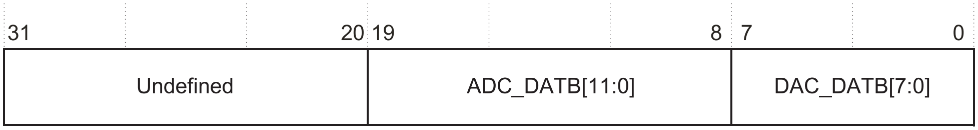


Figure 3-19 SYS_VOLTAGE1 register

Table 3-14 lists the SYS_VOLTAGE1 register bit assignments.

Table 3-14 SYS_VOLTAGE1 register

Bits	Access	Name	Reset value	Description
[31:20]	Read as zero, write ignored.	-	0x000	Undefined
[19:8]	Read only	ADC_DATB	0x000	Reads the **AVDD** voltage for the ARM11 MPCore test chip PLL. The LSB of the ADC reading corresponds to 610µV. The formula to calculate the supply voltage (AVDD) is: $AVDD=INT(ADC\_DATB[11:0]*(610*10^{-6}))$
[7:0]	Read/Write Lockable	DAC_DATB	0x80	Sets the **AVDD** voltage for the ARM11 MPCore test chip PLL. The default value is 1.2V. Circuit values are typically chosen to give a ±0.25V adjustment range for the PLL voltage. A value of 0xFF gives maximum negative offset from the default (-0.25V) and a value of 0x0 gives maximum positive offset from the default (+0.25V).

SYS_VOLTAGE2

This register is located at SYSBASE+0xA8 as follows:

Address [31:0]=0x100000A8

Figure 3-20 shows the SYS_VOLTAGE2 register bit assignments.

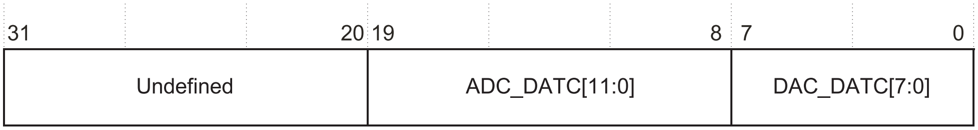


Figure 3-20 SYS_VOLTAGE2 register

Table 3-15 lists the SYS_VOLTAGE2 register bit assignments.

Table 3-15 SYS_VOLTAGE2 register

Bits	Access	Name	Reset value	Description
[31:20]	Read as zero, write ignored.	-	0x000	Undefined
[19:8]	Read Only	ADC_DATC	0x000	Reads the **VDDCORE** current for the MPCore. To calculate the current (IDDCORE) use the formula: $IDDCORE=INT(ADC\_DATC[11:0]*(610*10^{-6}))/R_{SENSE}*100$ By default, the R_{SENSE} resistor is 0.025Ω.
[7:0]	Reserved	DAC_DATC	0x00	Legacy only (not used).

SYS_VOLTAGE3

This register is located at SYSBASE+0xAC as follows:

Address [31:0]=0x100000AC

Figure 3-21 on page 3-59 shows the SYS_VOLTAGE3 register bit assignments.

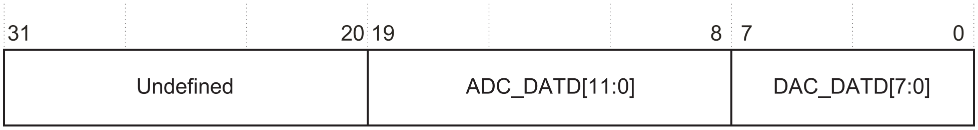


Figure 3-21 SYS_VOLTAGE3 register

Table 3-16 lists the SYS_VOLTAGE3 register bit assignments.

Table 3-16 SYS_VOLTAGE3 register

Bits	Access	Name	Reset value	Description
[31:20]	Read as zero, write ignored.	-	`0x000`	Undefined
[19:8]	Read Only	ADC_DATD	`0x000`	Reads the **AVDD** current for the ARM11 MPCore test chip PLL. To calculate the current (`AIDD`) use the formula: $AIDD=INT(ADC\_DATD[11:0]*(610*10^{-6}))/R_{SENSE}*100$ By default, the R_{SENSE} resistor is 0.5Ω.
[7:0]	Reserved	DAC_DATD	`0x00`	Legacy only (not used).

3.11 JTAG support

JTAG signals are present on both the upper and lower HDRZ connectors. An external board provides the JTAG connector and the routing of the JTAG signals from the connector to HDRZ (see *JTAG control and clock routing* on page 3-63). The Core Tile routes the JTAG scan path through devices on the board. The logic devices that are placed in the Core Tile scan chain depend on the JTAG mode:

Debug mode Debug mode is selected when the CONFIG slide-switch on the EB is set to OFF. It is the default mode used for general system development and debug. In this mode, the JTAG signals flow through the Debug Scan Chain (this scan chain connects to the ARM11 MPCore test chip only). The JTAG signals used for debug are identified by the **D_** prefix.

Note

Common access to the MPCore CPU DBGTAP controllers is available in this mode. See *Debug* on page 4-32.

Configuration mode

In configuration mode, all FPGAs, PLDs, clock generators, and the ARM11 MPCore test chip are placed into the scan chain. This mode allows the programmable logic devices in the system to be reprogrammed and the test chip registers to be configured.

To select configuration mode, set the CONFIG slide-switch on the EB to ON. This pulls the **nCFGEN** signal LOW on the Core Tile and reroutes the JTAG scan path. The JTAG signals used for configuration are identified by the **C_** prefix.

Note

Access to the ARM11 MPCore test chip boundary scan TAP controller is available in this mode. See *JTAG Configuration* on page 4-33.

Table 3-17 on page 3-61 provides a description of the JTAG signals.

Note

In the description in Table 3-17 on page 3-61, the term JTAG equipment refers to any hardware that can drive the JTAG signals to devices in the scan-chain. Typically, RealView ICE is used, although you can also use hardware from third-party suppliers to debug ARM processors.

Table 3-17 JTAG signal description

Name	Description	Function
nBSTAPEN	Boundary scan TAP enable	In configuration mode, the boundary scan TAP logic (if present) in a test chip is enabled. ——— **Note** ——— **nBSTAPEN** drives the **TESTMODE** control pin on the ARM11 MPCore test chip. In JTAG configuration mode, **nBSTAPEN** is HIGH and the ARM11 MPCore test chip TAP controller is enabled. In JTAG debug mode, **nBSTAPEN** is LOW and the four daisy-chained DBGTAP controllers in the MPCore are enabled. —————————
EDBGRQ[3:0]	External debug request (to CT11MPCore PLD)	**EDBGRQ[3:0]** is a request to the indexed MPCore CPU to enter the debug state.
DBGACK[3:0]	Debug acknowledge (to JTAG equipment)	**DBGACK[3:0]** indicates to the debugger that the indexed MPCore CPU has entered debug state.
nCFGEN	Configuration enable (controlled by config slide-switch on the baseboard)	**nCFGEN** is an active LOW signal used to put the boards into configuration mode. In configuration mode all FPGAs and PLDs are connected to the scan chain so that they can be configured by the JTAG equipment.
nRTCKEN	Return **TCK** enable (from Core Tile to baseboard)	**nRTCKEN** is an active LOW signal driven by any Core Tile that requires **RTCK** to be routed back to the JTAG equipment. If **nRTCKEN** is HIGH, the baseboard drives **RTCK** LOW. If **nRTCKEN** is LOW, the baseboard drives the **TCK** signal back up the stack to the JTAG equipment.
nTRST, **D_nTRST**, **C_nTRST**	Test reset (from JTAG equipment)	This active LOW open-collector signal is used to reset the JTAG port and the associated debug circuitry on the processor. It is asserted at power-up by each module, and can be driven by the JTAG equipment. This signal is also used in configuration mode to control the programming pin (**nPROG**) on FPGAs. **D_nTRST** is the reset for the debug mode scan chain and **C_nTRST** is the reset for the configuration mode scan chain. ——— **Note** ——— **D_nTRST** is always tied to **D_nSRST**. **C_nTRST** is tied to **D_nTRST** when configuration mode is enabled. —————————

Table 3-17 JTAG signal description (continued)

Name	Description	Function
RTCK, **D_RTCK**	Return TCK (to JTAG equipment)	Some devices sample **TCK** (for example a synthesizable core with only one clock), and this has the effect of delaying the time when a component actually captures data. **RTCK** is a mechanism for returning the sampled clock to the JTAG equipment, so that the clock is not advanced until the synchronizing device has captured the data. In a multiple device JTAG chain, the **D_RTCK** output from a component connects to the **TCK** input of the down-stream device. The **RTCK** signal on the EB connector HDRZ returns **TCK** to the JTAG equipment. **D_RTCK** is the **RTCK** signal in the debug scan chain. **RTCK** is not available in the configuration mode scan chain.
TCK, **D_TCK**, **C_TCK**	Test clock (from JTAG equipment)	**TCK** synchronizes all JTAG transactions. **TCK** connects to all JTAG components in the scan chain. Series termination resistors are used to reduce reflections and maintain good signal integrity. **TCK** flows up the stack of modules and connects to each JTAG component. However, if there is a device in the scan chain that synchronizes **TCK** to some other clock, then all down-stream devices are connected to the **RTCK** signal on that component (see **RTCK**). **D_TCK** is the clock for the debug mode scan chain and **C_TCK** is the clock for the configuration mode scan chain.
TDI, **D_TDI**, **C_TDI**	Test data in (from JTAG equipment)	**TDI** goes up the stack of tiles from the baseboard (or Interface Module) and then back down the stack (as **TDO**) connecting to each component in the scan chain. **D_TDI** is the data signal for the debug mode scan chain and **C_TDI** is the data signal for the configuration mode scan chain.
TDO, **D_TDO**, **C_TDO**	Test data out (to JTAG equipment)	**TDO** is the return path of the data input signal **TDI**. For a stack of RealView products, **TDI** goes up to the top of the stack and returns down as **TDO**. The JTAG components are connected in the return path so that the length of track driven by the last component in the chain is kept as short as possible. **D_TDO** is the data signal for the debug mode scan chain and **C_TDO** is the data signal for the configuration mode scan chain.
TMS, **D_TMS**, **C_TMS**	Test mode select (from JTAG equipment)	**TMS** controls transitions in the tap controller state machine. **TMS** connects to all JTAG components in the scan chain. **D_TMS** is the control signal for the debug mode scan chain and **C_TMS** is the control signal for the configuration mode scan chain.

3.11.1 JTAG control and clock routing

Figure 3-22 shows the JTAG *Debug Mode* routing of the **D_nTRST**, **D_TCK**, and **D_TMS** signals to the ARM11 MPCore test chip. The bus switches, unless otherwise indicated, are controlled by **nCFGEN** from the baseboard. JTAG debug mode routing is selected when **nCFGEN** is HIGH. This is the default baseboard setting.

Note

At power-up, the baseboard **nSYSPOR** power-on-reset signal initializes the boundary scan tap controller and the four DBGTAP controllers in the ARM11 MPCore test chip by forcing **ARM_nTRST** LOW.

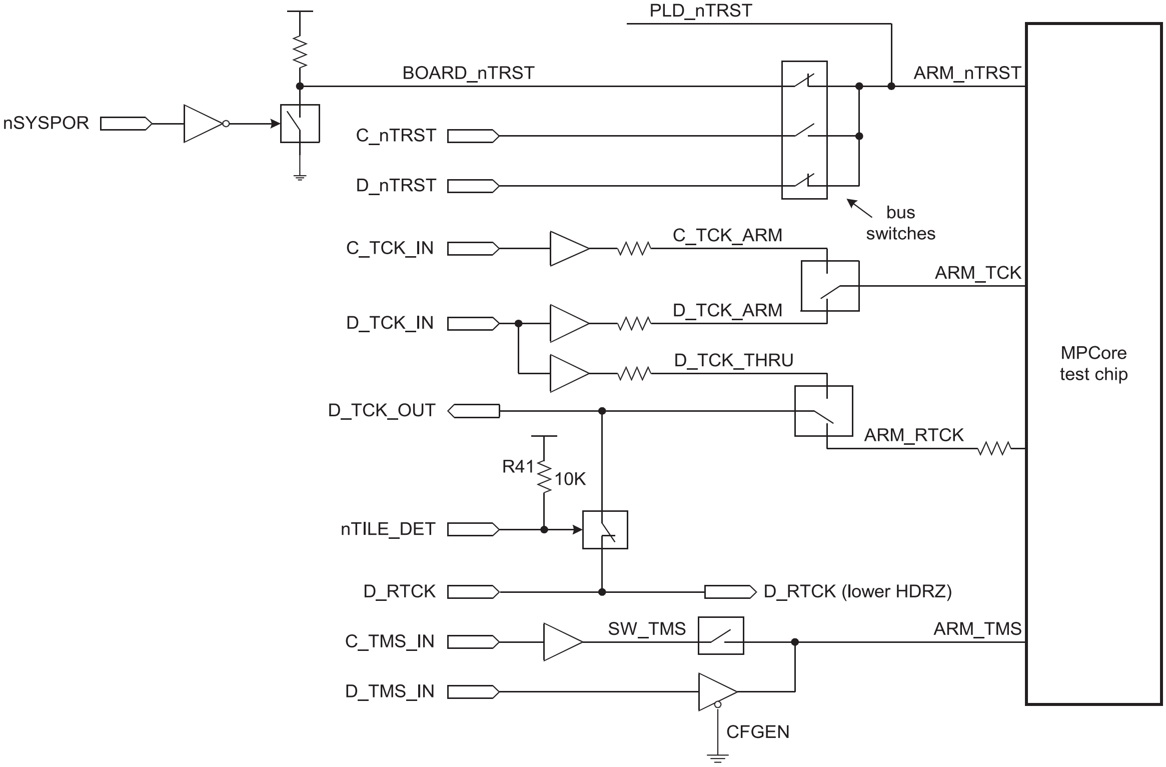


Figure 3-22 JTAG debug mode routing (nCFGEN=1)

The JTAG input and output routing shown in Figure 3-22 on page 3-63 is for a single tile. In this case, no tile is connected to the upper HDRZ header and **D_TCK_OUT**, and **D_RTCK** from the upper HDRZ header, are not used. For details on JTAG input and output routing in a tile stack, see *JTAG loopback control* on page 3-68. JTAG loopback routing is controlled by **nTILE_DET**.

Figure 3-23 shows the JTAG *Configuration Mode* routing of the **C_nTRST**, **C_TCK**, and **C_TMS** signals to the ARM11 MPCore test chip. The bus switches, unless otherwise indicated, are controlled by **nCFGEN** from the baseboard. JTAG configuration mode routing is selected when **nCFGEN** is LOW.

Note

To select JTAG configuration mode, set the CONFIG slide-switch to ON on the EB. This pulls the **nCFGEN** signal LOW on the CT11MPCore and reroutes the JTAG scan path.

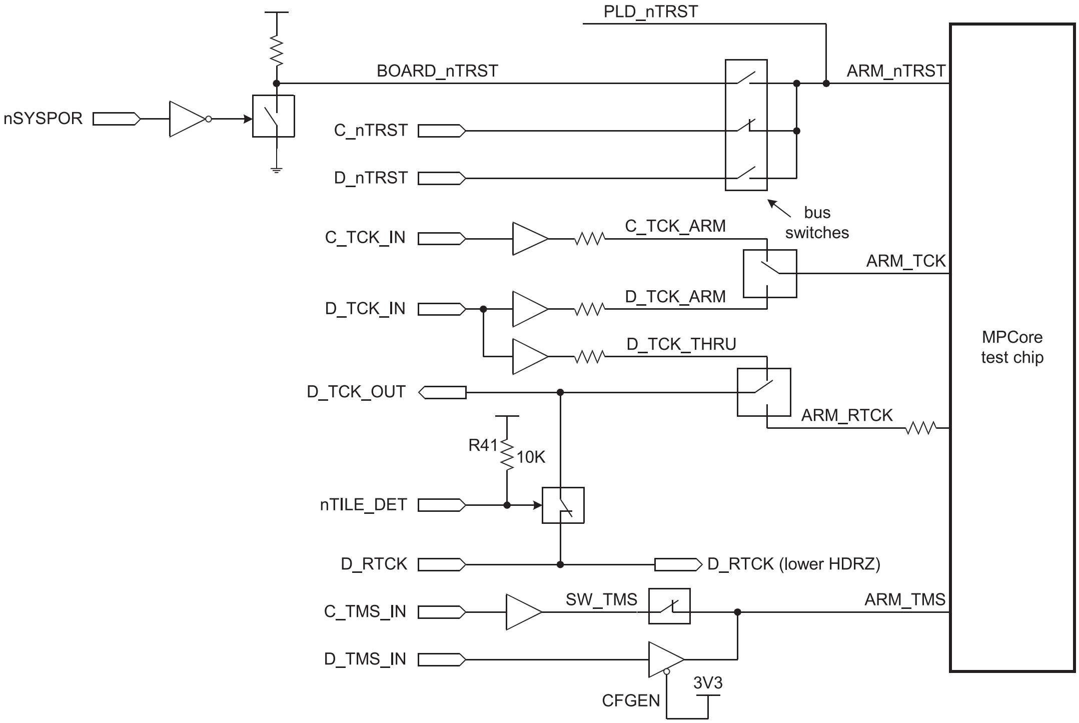


Figure 3-23 JTAG configuration mode routing (nCFGEN=0)

The Core Tile JTAG input and output routing shown in on page 3-64 is for a single tile. In this case, no tile is connected to the upper HDRZ header and **D_TCK_OUT**, and **D_RTCK** from the upper HDRZ header, are not used. For details on Core Tile JTAG input and output routing in a tile stack, see *JTAG loopback control* on page 3-68. JTAG loopback routing is controlled by **nTILE_DET**.

Note

PLL_nTRST is tied to **ARM_nTRST**. See Figure 3-23 on page 3-64 for details.

The ispClock generator does not have a **nTRST** pin. The TAP controller is reset by keeping **TMS** HIGH, this forces its TAP controller into the *Test-Logic-Reset* steady state within 5 **TCK** cycles. *Test-Logic-Reset* is also the power-on default state.

Figure 3-24 shows the distribution of the critical **C_TCK** and **C_TMS** signals to the remaining JTAG configurable devices on the CT11MPCore. Buffered versions of the signals, **C_TCK_OUT** and **C_TMS_OUT** are provided at the upper HDRZ header to drive the **C_TCK** and **C_TMS** inputs of the next tile in a tile stack.

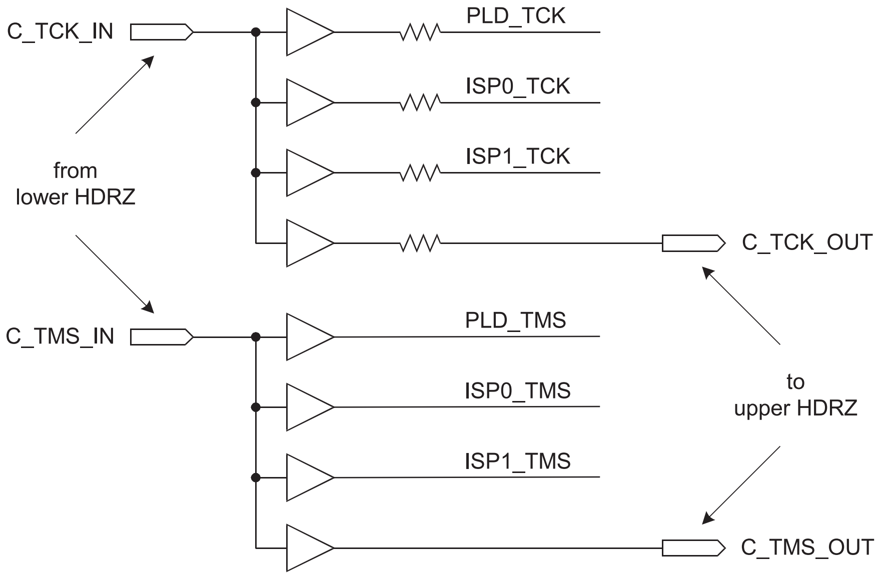


Figure 3-24 Additional JTAG configuration mode routing

3.11.2 JTAG debug mode scan chain routing

Figure 3-25 on page 3-66 shows the JTAG *Debug Mode* scan chain routing.

In JTAG debug (normal) mode the ARM11 MPCore test chip is connected in the debug scan chain. The bus switches, unless otherwise indicated, are controlled by **nCFGEN** from the baseboard. JTAG debug mode routing is selected when **nCFGEN** is HIGH. This is the default baseboard setting.

The JTAG input and output routing shown in Figure 3-25 is for a single tile. In this case, no tile is connected to the upper HDRZ header and **D_TDO_IN** and **C_TDO_IN** are not used. For details on JTAG input and output routing in a tile stack, see *JTAG loopback control* on page 3-68. JTAG loopback routing is controlled by **nTILE_DET**.

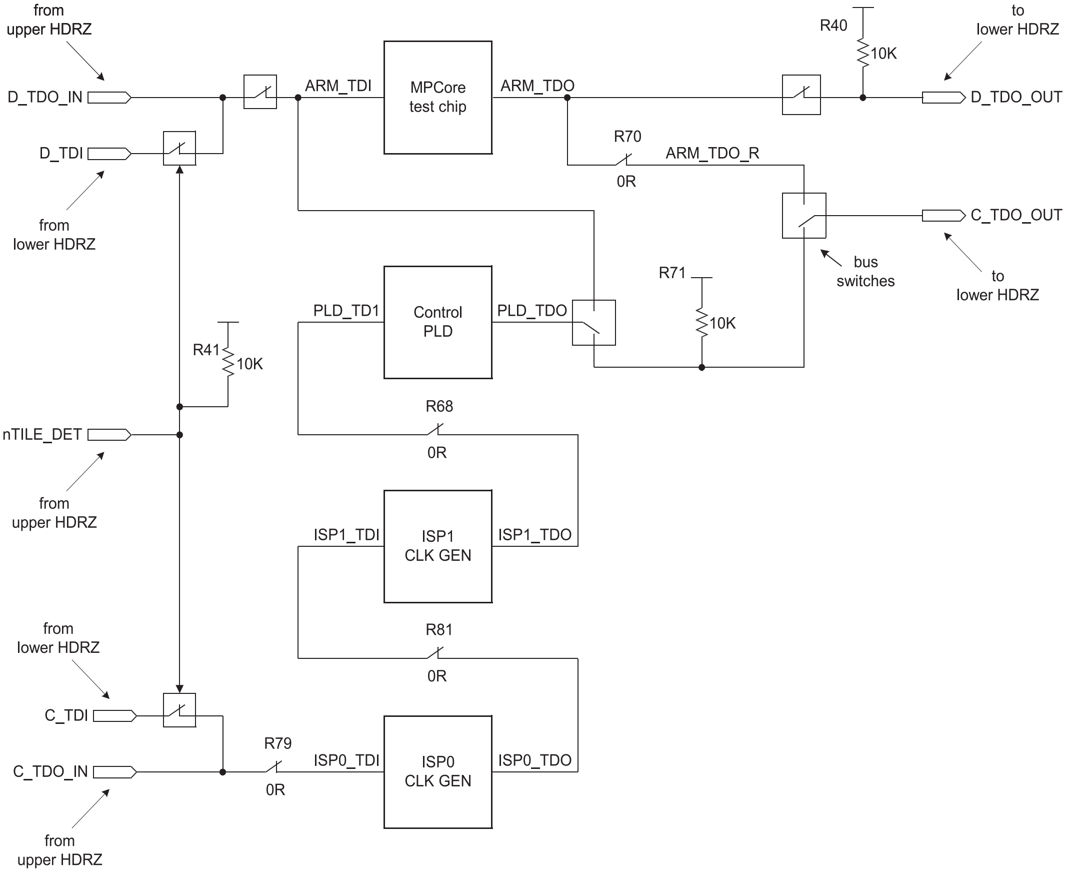


Figure 3-25 JTAG debug (normal) mode scan chain routing (nCFGEN=1)

3.11.3 JTAG configuration mode scan chain routing

Figure 3-26 shows the JTAG *Configuration Mode* routing.

In JTAG configuration mode all the JTAG configurable devices (ISP clock generator 0, ISP clock generator 1, PLD, and ARM11 MPCore test chip) are connected in the configuration scan chain. The bus switches are controlled by **nCFGEN** from the baseboard. JTAG configuration mode routing is selected when **nCFGEN** is LOW.

The JTAG input and output routing shown in Figure 3-26 is for a single tile. In this case, no tile is connected to the upper HDRZ header and **D_TDO_IN** and **C_TDO_IN** are not used. For details on JTAG input and output routing in a tile stack, see *JTAG loopback control* on page 3-68. JTAG loopback routing is controlled by **nTILE_DET**.

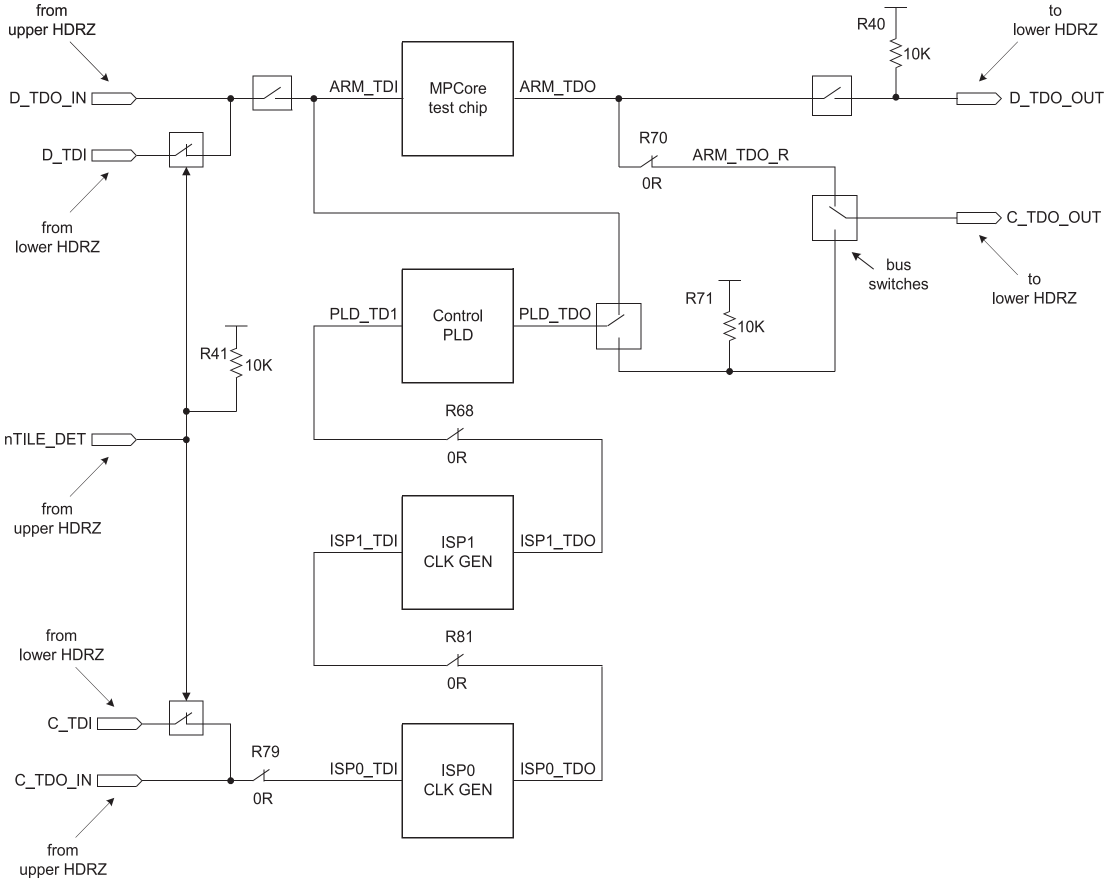


Figure 3-26 JTAG configuration mode scan chain (nCFGEN=0)

3.11.4 JTAG loopback control

A number of tiles may be stacked above the EB. Figure 3-27 shows the TDI, TDO, and RTCK loopback control for a multi-tile stack. The JTAG debug and configuration TDI signals from the baseboard, **D_TDI** and **C_TDI** are made available to every tile in the stack by feeding through from the lower to the upper header HDRZ on each tile. The bus switches are controlled by **nTILE_DET**. If a tile is the top tile in the stack **nTILE_DET** remains pulled HIGH. **C_TDI** and **D_TDI** loopback is then achieved by the bus switches connecting this last tile in the stack to the **C_TDI** and **D_TDI** signals. For tiles further down the stack **nTILE_DET** is pulled LOW by the tile above and these tiles connect to the **C_TDO_OUT** and **D_TDO_OUT** signals from the tile above.

D_RTCK is made available from every tile in the stack by feeding through from the upper to the lower header HDRZ on each tile. The bus switches are controlled by **nTILE_DET**. If a tile is the top tile in the stack **nTILE_DET** remains pulled HIGH. **D_RTCK** loopback is then achieved by the bus switches connecting **D_TCK_OUT** from this last tile in the stack to **D_RTCK**. For tiles further down the stack **nTILE_DET** is pulled LOW by the tile above and these tiles connect to the **D_TCK_OUT** signal from the tile below.

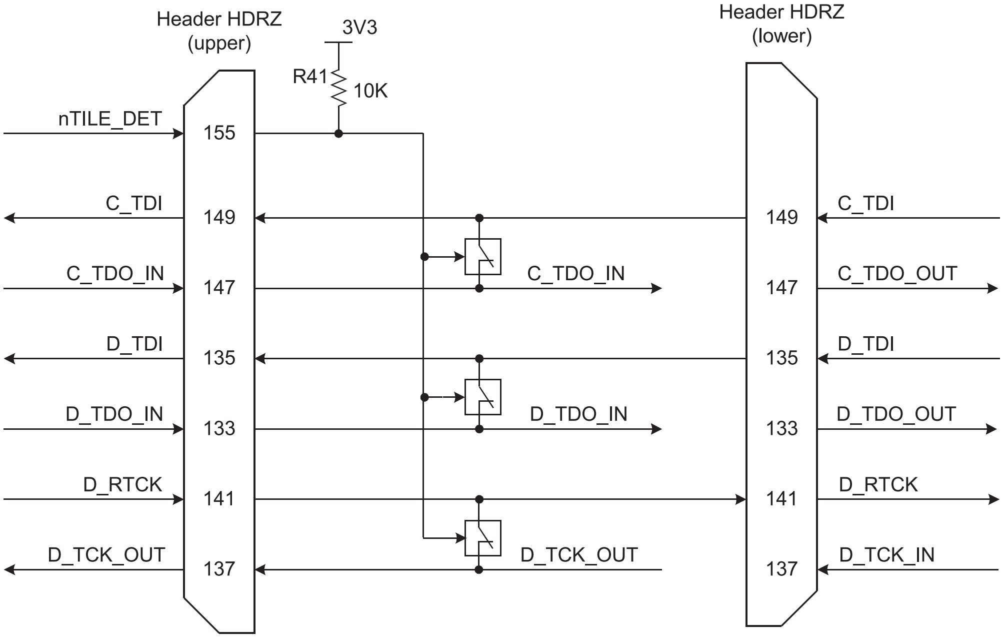


Figure 3-27 JTAG loopback control

3.11.5 Debug request and acknowledge routing

Routing of the ARM11 MPCore external debug request and debug acknowledge signals, **EDBGRQ[3:0]** and **DBGACK[3:0]** is not implemented in the CT11MPCore PLD image provided with this CD release. This will be implemented in future releases of this product.

Chapter 4
Test Chip Hardware Description

This chapter provides details of the ARM11 MPCore test chip which affect the CT11MPCore.

It contains the following sections:

——— **Note** ———

Refer to the *ARM11 MPCore Processor Technical Reference Manual* and the *L220 Cache Controller Technical Reference Manual* for further details on the major test chip components.

———————————

4.1 ARM11 MPCore test chip overview

Figure 4-1 shows the top-level functionality of the test chip.

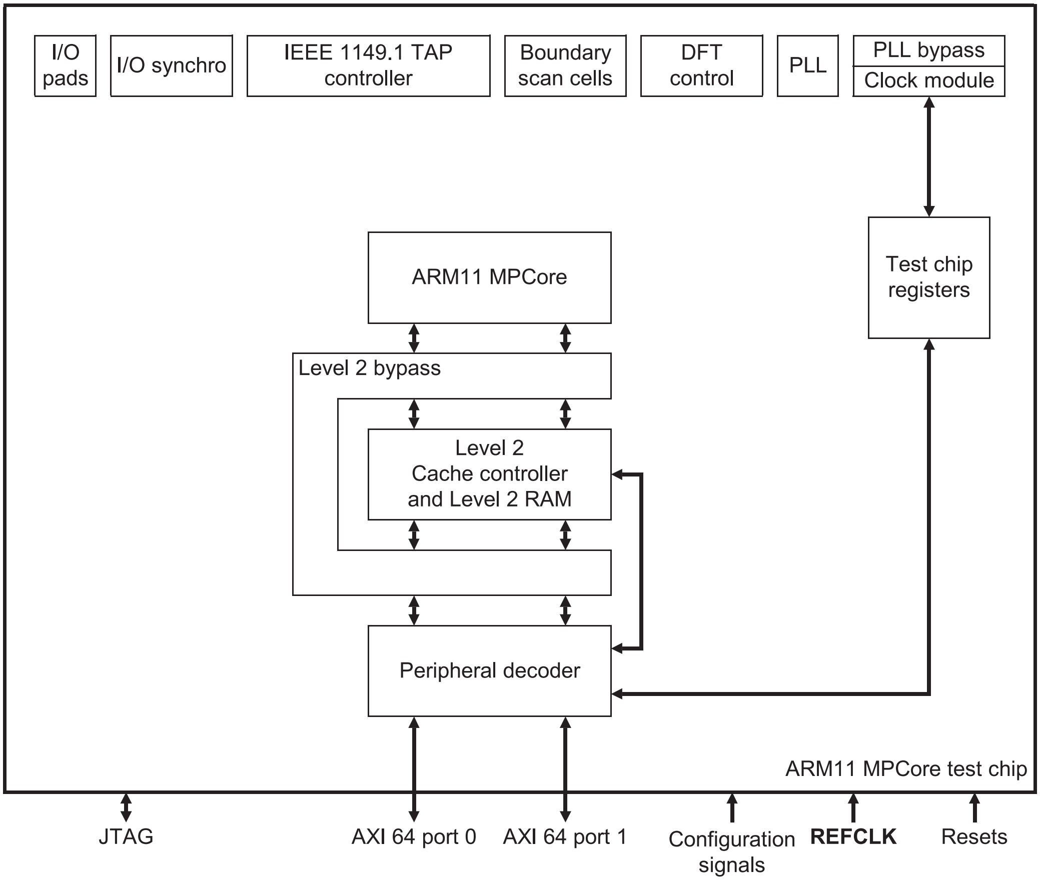


Figure 4-1 Top-level view of ARM11 MPCore test chip

The main features of the test chip are:

- a ARM11 MPCore incorporating:
  - four ARM11 MPCore CPUs with Vector Floating Point (VFP) that implement the ARM architecture v6
  - Level 1 (L1) memory subsystem providing 32KB instruction cache and 32KB data cache per CPU
  - JTAG-based debug.

- L220 Cache Controller incorporating 1MB of Level 2 (L2) unified cache
- peripheral decoder providing dual 64-bit external AXI buses
- on-chip PLL with skew control
- test chip register bank for configuring the ARM11 MPCore test chip
- design for test (DFT):
  - test control via a TEST_MODE test data register
  - L1 and L2 memory systems Memory Built-In Self Test (MBIST)
  - full chip and boundary scan support

See the *ARM11 MPCore Processor Technical Reference Manual* and the *L220 Cache Controller Technical Reference Manual* for further details on these features.

4.2 Clocks

This section describes the clocking requirements for the ARM11 MPCore test chip and contains the following subsections:

- *Clocking overview*
- *PLL* on page 4-6
- *PLL bypass module* on page 4-7
- *Test chip PLL control register* on page 4-8
- *Clock module* on page 4-10
- *Test chip clock divider register* on page 4-11
- *Clock signals overview* on page 4-12.

4.2.1 Clocking overview

Figure 4-2 on page 4-5 shows the clock domains within the ARM11 MPCore test chip.

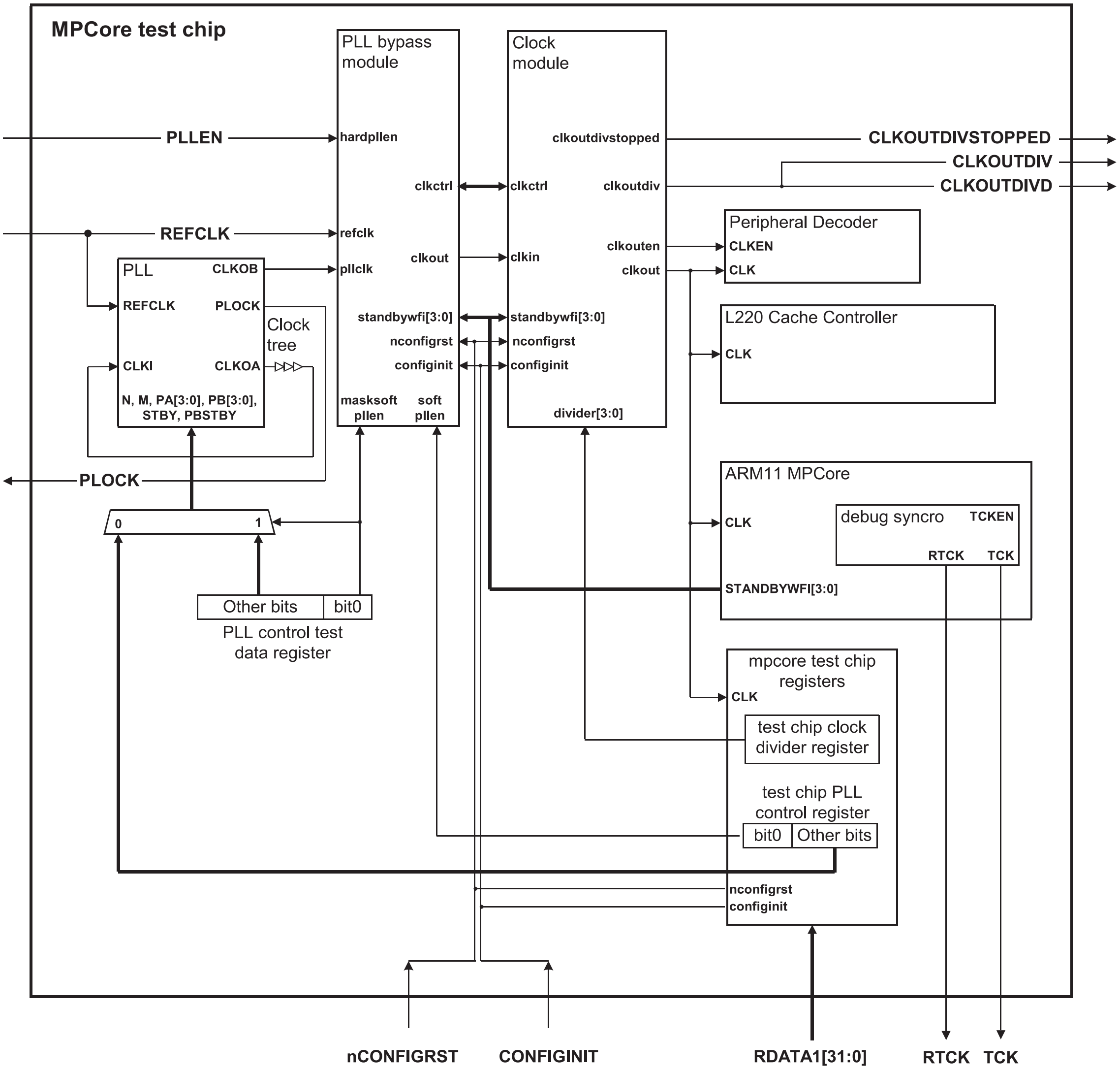


Figure 4-2 Clock domain overview

The MPCore test chip clock domain consists of the PLL, the PLL bypass module, and the Clock module. As well as generating the main clock, **CLK** the PLL bypass module and the Clock module enable the bypassing of the PLL so that the test chip input clock,

REFCLK can be used to generate the internal and external clocks. The PLL bypass module ensures that the Clock module can switch from the PLL output clock, **CLKOB** to the test chip input clock, **REFCLK** without glitches.

ARM11 MPCore

The ARM11 MPCore has a single clock domain and is clocked by the main clock, **CLK**. generated by the test chip Clock module.

The debug signals **TCK**, **RTCK**, and **TCKEN** are generated in the ARM11 MPCore. **TCK** is synchronized to the main clock, **CLK** in the ARM11 MPCore.

L220 Cache Controller

Clocking of the L220 RAM arrays is synchronized to the main clock, **CLK** by the L220.

Peripheral decoder

The Peripheral decoder is required to interface externally at a lower AXI bus frequency. A clock enable signal, **ACLKEN** combined with the main clock, **CLK** are used to drive this interface. **CLK** and **ACLKEN** are generated by the test chip Clock module.

ARM11 MPCore test chip registers

The test chip registers are clocked by the main clock, **CLK** generated by the test chip Clock module.

4.2.2 PLL

The ARM11 MPCore test chip contains an on-board PLL with skew control. Skew control ensures there is a known fixed phase delay between the feedback clock to the PLL and the reference clock, **REFCLK**. Phase delay is typically 0ns, with a jitter in the order of 100ps.

The main clock required by the ARM11 MPCore test chip is provided by either:

- the reference clock, **REFCLK** from an external source
- the integrated PLL, **CLKOB** output.

The feedback clock, for skew control, must always be **CLKOA**. Therefore, the clock tree insertion delay is always put in the **CLKOA** path. The skew between **CLKOA** and **CLKOB** is very small and is in the order of 105ps. Figure 4-3 on page 4-7 shows the PLL block diagram.

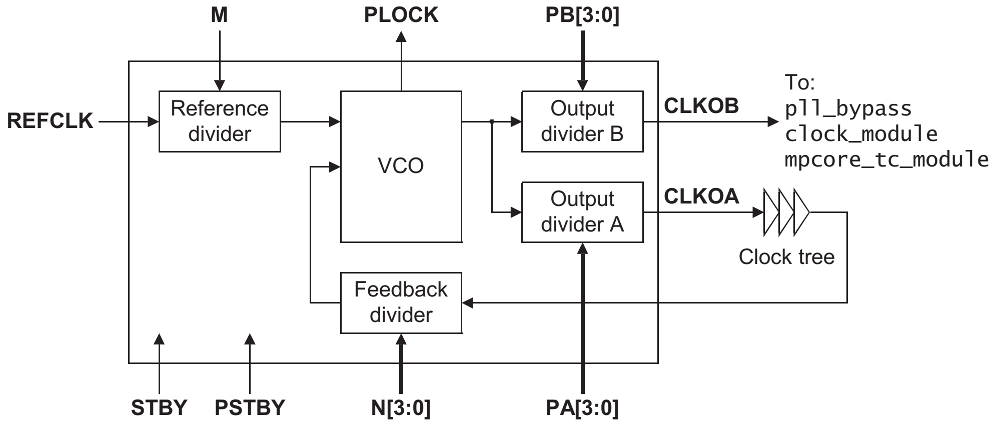


Figure 4-3 PLL block diagram

The following equations describe the relationships between **REFCLK**, **CLKOA**, **CLKOB**, **Fvco**, pa-value, pb-value, m-value, and n-value.

$$\frac{\textbf{REFCLK}}{\text{m-value}} = \frac{\textbf{Fvco}}{\text{pa-value x n-value}} \quad \text{therefore,} \quad \textbf{Fvco} = \frac{\textbf{REFCLK} \text{ x pa-value x n-value}}{\text{m-value}}$$

$$\textbf{CLKOA} = \frac{\textbf{Fvco}}{\text{pa-value}} \quad \text{therefore,} \quad \textbf{CLKOA} = \frac{\textbf{REFCLK} \text{ x n-value}}{\text{m-value}}$$

$$\textbf{CLKOB} = \frac{\textbf{Fvco}}{\text{pb-value}} \quad \text{therefore,} \quad \textbf{CLKOB} = \frac{\textbf{REFCLK} \text{ x pa-value x n-value}}{\text{m-value x pb-value}}$$

Referring to the *PLL block diagram*:

- the n-value comes from **N[3:0]** and can have a value between 1 and 16
- the m-value comes from **M** and can have a value of 1, or 2
- the pa-value comes from **PA[3:0]** and can have a value between 1 and 16
- the pb-value comes from **PB[3:0]** and can have a value between 1 and 16.

4.2.3 PLL bypass module

The PLL bypass module enables you to select between **REFCLK** and **CLKOB** as output. These signals are considered to be unrelated and asynchronous. Logic is included in the test chip to prevent glitches when the selected clock changes.

Note

Any pulse, HIGH or LOW, with a duration shorter than the corresponding pulse of the input clock, before or after the selected clock switches, is considered to be a glitch.

To ensure glitch free clock switching:

- the selection of the previous clock must go from HIGH to LOW before the selection of the new clock goes from LOW to HIGH
- A guard band time must be provided between deselecting and selecting the clocks to guarantee there is no overlap. This is because the clocks have different frequencies and synchronizations. The guard band time is set by using a timer. The lowest frequency, **REFCLK**, drives the timer, and the timer counts four cycles whenever the selection changes.

Note

REFCLK is considered to be the lowest clock frequency, so it must always be lower than **CLKOB**. You must take this into account when you program the PLL divider values.

Either of the following can bypass the PLL:

- a hardware PLL enable, **PLLEN**, an external primary input pin
- bit 0, of the test chip PLL control register, a software PLL enable.

You can initialize the *Test chip PLL control register* before you release the main reset by using the test chip **CONFIGINIT** feature. This is the initialization method used by the system FPGA on the EB. See *Clock setting in reset mode* on page 3-6 for further information on this hardware initialization feature.

4.2.4 Test chip PLL control register

A 32-bit read/write register programs several **REFCLK/CLKOB/CLKOA** ratios that the PLL requires. This register contains a bit to enable/disable the PLL using software. Use this register in a read-modify-write sequence, so that you do not disturb any other ratios. You can also access this PLL control register after releasing the main reset using an AXI interface connected to the Peripheral decoder. See *Peripheral accesses* on page 3-45 for details.

This register is located at first address PERIPHBASE + 12KB as follows:

```
Address [31:14]=PERIPHBASE [17:0]
Address [13:12]=b11, Address [11:0]=0x00
```

Figure 4-4 on page 4-9 shows the test chip PLL control register bit assignments.

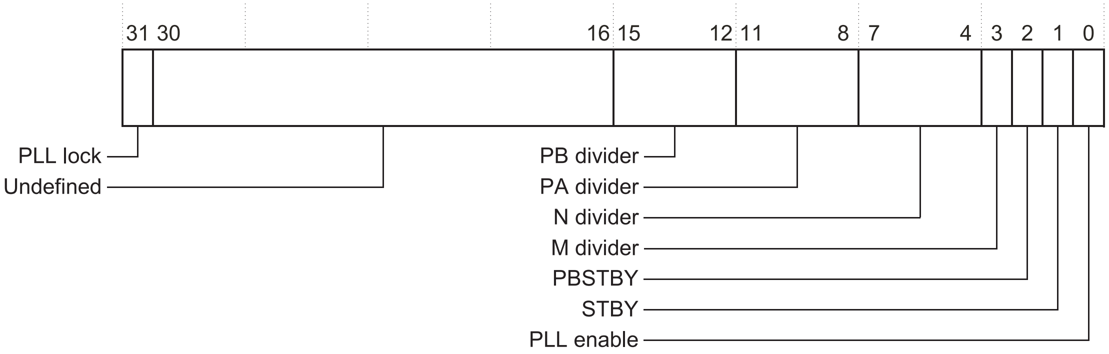


Figure 4-4 Test chip PLL control register

Table 4-1 lists the test chip PLL control register bit assignments.

Table 4-1 Test chip PLL control register

Bits	Read/write	Name	Reset value	Description
[31]	RO	PLL lock	`b0`	Reflects the **PLOCK** output of the PLL: • 0: PLL is not locked • 1: PLL is locked.
[30:16]	Read as zero, write ignored.	-	`b0`	Undefined
[15:12]	RW	PB divider	`b0000`	PB divider ratio that the PLL uses: 0000: pb-value is 1 0001: pb-value is 2 … 1111: pb-value is 16.
[11:8]	RW	PA divider	`b0000`	PA divider ratio that the PLL uses: 0000: pa-value is 1 0001: pa-value is 2 … 1111: pa-value is 16.
[7:4]	RW	N divider	`b0000`	N divider ratio that the PLL uses: 0000: n-value is 1 0001: n-value is 2 … 1111: n-value is 16.

Table 4-1 Test chip PLL control register (continued)

Bits	Read/write	Name	Reset value	Description
[3]	RW	M divider	`b0`	M divider ratio that the PLL uses: • 0: m-value is 1 • 1: m-value is 2.
[2]	RW	PBSTBY	`b1`	• 0: **CLKB** from PLL is active • 1: **CLKB** from PLL is 0.
[1]	RW	STBY	`b1`	• 0: **CLKA** from PLL is active • 1: **CLKA** from PLL is 0.
[0]	RW	PLL enable	`b0`	• 0: if *committed* and bit 0 of the PLL control test data register is 0 (its reset value) **REFCLK** will be selected as **CLKOUT** on the next rising edge of **REFCLK** • 1: if *committed*, and bit 0 of the PLL control test data register is 0 (its reset value) the PLL will be selected as **CLKOUT** on the next rising edge of **REFCLK**. PLL enable is *committed* when **STANDBYWFI[3:0]** == b1111 or the input signal **CONFIGINIT** == b1.

Note

Because the **REFCLK** frequency must always be lower than **CLKOB**, to enable normal behavior of the PLL bypass module, then the following must be true:

(n-value × pa-value)/(m-value × pb-value) ≥ 1

4.2.5 Clock module

Figure 4-2 on page 4-5 shows the Clock module inputs and outputs. The **clkout_o** signal is exactly **clkin_i**, and **clkoutdiv_o** is a divided version of **clkin_i**. An output signal, **clkouten_o**, enables you to use **clkout_o** either in combination with **clkouten_o**, or in combination with **clkoutdiv_o** as Figure 4-5 on page 4-11 shows.

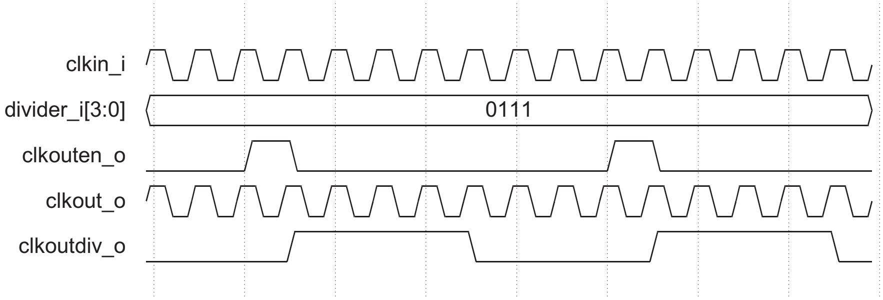


Figure 4-5 AXI bus clock

Note

- The **clkout_o** and **clkouten_o** signals drive the internal AXI components.
- The **clkoutdiv_o** signal drives the external AXI components.

The clock divider value is set by the **divider_i[3:0]**output from the *Test chip clock divider register*.

The *test chip clock divider register* can be initialized before releasing the main reset by using the test chip **CONFIGINIT** feature. This is the initialization method used by the *CT11MPCore PLD*. See *Clock setting in reset mode* on page 3-6 for a description this hardware initialization feature. You can also access this test chip divider register after releasing the main reset using an AXI interface connected to the Peripheral decoder. See *Peripheral accesses* on page 3-45 for details.

4.2.6 Test chip clock divider register

The test chip clock divider register is located at address PERIPHBASE + 12KB as follows:

```
Address [31:14]=PERIPHBASE [17:0]
Address [13:12]=b11, Address [11:0]=0x014
```

Figure 4-6 on page 4-12 shows the test chip clock divider register bit assignments.

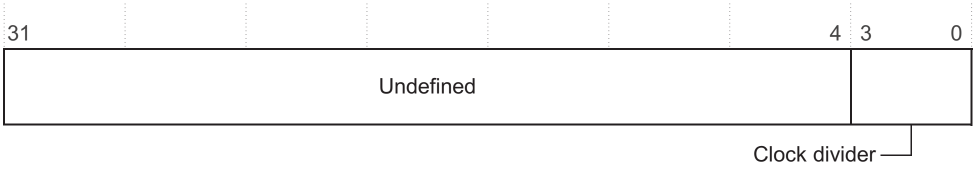


Figure 4-6 Test chip clock divider register

Table 4-2 lists the test chip clock divider register bit assignments.

Table 4-2 Test chip clock divider register

Bits	Read/write	Name	Reset value	Description
[31:4]	Read as zero, write ignored	-	`b0`	Undefined
[3:0]	RW	Clock divider	`b0000`	The clock divider ratio to drive the AXI bus clock and the FPGA: 0000: **CLKOUTDIV** is equal to **CLKIN** 0001: **CLKOUTDIV** is **CLKIN** divided by 2 0010: **CLKOUTDIV** is **CLKIN** divided by 3 0011: **CLKOUTDIV** is **CLKIN** divided by 4 0100: **CLKOUTDIV** is **CLKIN** divided by 5 0101: **CLKOUTDIV** is **CLKIN** divided by 6 0110: **CLKOUTDIV** is **CLKIN** divided by 7 0111: **CLKOUTDIV** is **CLKIN** divided by 8 1000: **CLKOUTDIV** is **CLKIN** divided by 1 1001: **CLKOUTDIV** is **CLKIN** divided by 10 1010: **CLKOUTDIV** is **CLKIN** divided by 3 1011: **CLKOUTDIV** is **CLKIN** divided by 12 1100: **CLKOUTDIV** is **CLKIN** divided by 5 1101: **CLKOUTDIV** is **CLKIN** divided by 14 1110: **CLKOUTDIV** is **CLKIN** divided by 7 1111: **CLKOUTDIV** is **CLKIN** divided by 16

4.2.7 Clock signals overview

The clock related signals are summarized in Table 4-3 on page 4-13.

Table 4-3 Clock signals

Signal	Direction	Description	Toggling rate	Pad type
CLKOUTDIV	Output	This is the AXI bus clock. You can use it to drive a companion FPGA.	18.75-133MHz	9mA output pad
CLKOUTDIVD	Output	Reflects exactly the **CLKOUTDIV** output. This is a second pad to drive out the AXI bus clock.	18.75-133MHz	9mA output pad
CLKOUTDIVSTOPPED	Output	When HIGH, indicates that the **CLKOUTDIV** output becomes LOW.	18.75-133MHz	9mA output pad
CONFIGINIT	Input	Clock enable according to **REFCLK**. The PLL and clock divider control registers use this clock enable.	18.75-133MHz	Standard input pad
nCONFIGRST	Input	Resets the PLL and clock divider control registers.	Not applicable	Schmidt trigger input pad
PLLEN	Input	Enables or disables the PLL. When LOW, **REFCLK** drives the entire test chip.	Static input	Standard input pad
PLOCK	Output	Indicates whether the PLL is locked. 0: PLL is not locked. 1: PLL is locked	18.75-133MHz	9mA output pad
REFCLK	Input	Global clock. This clock is a reference for the PLL.	18.75-133MHz	Schmidt trigger input pad

4.3 Resets and interrupts

This section describes the reset and interrupt signals that enter the ARM11 MPCore test chip pins and contains the following sections:

- *Resets*
- *Interrupts* on page 4-17.

4.3.1 Resets

This section describes all the reset signals that enter the test chip:

nTRST This is the test logic reset. It also acts as the reset to the TAP controller.

nSYSPORESET

This is the main system reset on the ARM11 MPCore test chip. It resets:

- The peripheral decoder.
- The test chip-specific registers, apart from the test chip PLL control register, and the test chip clock divider register.
- The L220.
- All the debug registers that can be reset, and all the watch and breakpoint registers in the ARM11 MPCore. It also drives the ARM11 MPCore snoop control unit **nSCURESET** signal.
- **MBISTRESETN** of the ARM11 MPCore MBIST controller.
- **MBISTRESETN** of the L220 MBIST controller.

nCONFIGRST

This signal resets:

- the test chip PLL control register
- the test chip clock divider register
- the PLL bypass module
- the clock module.

Note

The CT11MPCore uses the **CONFIGINIT** feature and activates this signal before all other reset signals. See *Clock setting in reset mode* on page 3-6 for more information.

nWDRESET[3:0]

Individual watchdog resets inputs.

nCPURESET[3:0]

These signals individually reset each ARM11 MPCore processor by waking up processors that are in WFI state.

nPORESET[3:0]

Drives the ARM11 MPCore **nPORESET[3:0]** inputs.

RESETREQ[3:0]

This is an output to request individual watchdog resets.

Table 4-4 describes how the resets are handled.

Table 4-4 Reset handling

nTRST	nCONFIGRST	nSYSPORESET	nPORESET[3:0]	nWDRESET[3:0]	nCPURESET[3:0]	Description
X	0	X	X	X	X	Resets: • test chip PLL control register • test chip clock divider register • PLL bypass module • clock module.
X	X	0	All 0	All 0	All 0	Power-on reset. Resets: • all processors • Snoop Control Unit (SCU) • interrupt distributor • test chip logic.
X	X	1	[n]=0	[n]=0	[n]=0	Individual processor power-on reset. Resets all processor logic, including debug logic.
X	X	1	All 1	[n]=0	[n]=0	Individual processor soft reset. Resets all processor logic, excluding debug logic.

Table 4-4 Reset handling (continued)

nTRST	nCONFIGRST	nSYSPORESET	nPORESET[3:0]	nWDRESET[3:0]	nCPURESET[3:0]	Description
X	X	1	All 1	All 1	[n]=0	Individual processor reset from watchdog request, processor logic reset, excluding debug and watchdog reset flag.
0	X	X	X	X	X	Resets the TAP logic embedded in the ARM11 MPCore. Generates and synchronizes: • **TDI** • **TMS** • **TCK** • **RTCK** • test chip TAP controller.
X	X	1	All 1	All 1	All 1	No reset, normal run mode.

Note

Ensure that you assert all reset signals asynchronously, and deassert them synchronously with respect to their clock domains.

When in test mode, **TESTMODE** is tied to Vdd. When the test mode selected enables the SCAN, all resets must be asynchronous. Table 4-5 lists the reset deassertions with the respective clock.

Table 4-5 Reset deassertion with respective clock

Reset signal	Deasserted synchronously with primary input:	Deasserted synchronously with Clock module output:
nTRST	**TCK**	**clkout_o** when **clkouten_o** is HIGH
nCONFIGRST	**REFCLK**	**clkout_o**
nSYSPORESET	N/A	**clkout_o** when **clkouten_o** is HIGH

Table 4-5 Reset deassertion with respective clock (continued)

Reset signal	Deasserted synchronously with primary input:	Deasserted synchronously with Clock module output:
nWDRESET[3:0]	N/A	**clkout_o** when **clkouten_o** is HIGH
nCPURESET[3:0]	N/A	**clkout_o** when **clkouten_o** is HIGH
nPORESET[3:0]	N/A	**clkout_o** when **clkouten_o** is HIGH

4.3.2 Interrupts

This section describes all the interrupt signals that enter the test chip:

The ARM11 MPCore test chip has the following interrupt lines:

nIRQ[3:0] CPU legacy IRQ request input lines. Each bit is connected directly to each corresponding CPU:

nIRQ [0] to CPU#0
nIRQ [1] to CPU#1
nIRQ [2] to CPU#2
nIRQ [3] to CPU#3

nFIQ[3:0] CPU private FIQ request input lines. Each bit is connected directly to each corresponding CPU:

nFIQ [0] to CPU#0
nFIQ [1] to CPU#1
nFIQ [2] to CPU#2
nFIQ [3] to CPU#3

You can configure each *Fast Interrupt Request* (FIQ) as a *Non Maskable Interrupt* (NMI) by using a specific test chip register, the test chip interrupt control register. Figure 4-7 on page 4-18 shows the test chip interrupt control register bit assignments.

This register is located at address:

```
Address [31:14]=PERIPHBASE [17:0]
Address [13:12]=11, Address [11:0]=0x004
```

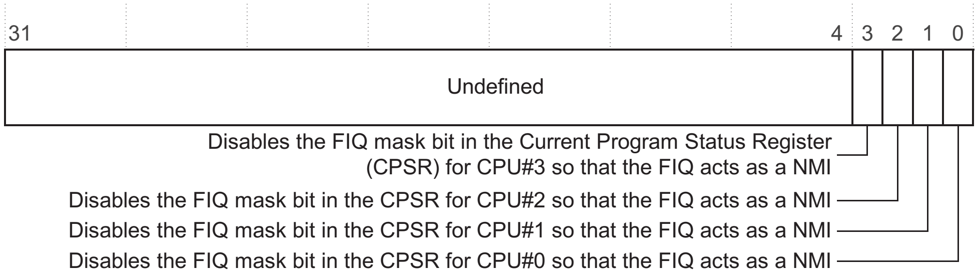

31
4
3
2
1
0
Undefined
Disables the FIQ mask bit in the Current Program Status Register (CPSR) for CPU#3 so that the FIQ acts as a NMI
Disables the FIQ mask bit in the CPSR for CPU#2 so that the FIQ acts as a NMI
Disables the FIQ mask bit in the CPSR for CPU#1 so that the FIQ acts as a NMI
Disables the FIQ mask bit in the CPSR for CPU#0 so that the FIQ acts as a NMI

Figure 4-7 Test chip interrupt control register

Table 4-6 describes the test chip interrupt control register.

Table 4-6 Test chip interrupt control register

Bits	Read/write	Reset value	Description
[31:4]	Read as zero, write ignored	0	Undefined
[3]	RW	0	Disable the FIQ mask bit in the *Current Program Status Register* (CPSR) for CPU#3 so that the FIQ acts as a NMI
[2]	RW	0	Disable the FIQ mask bit in the CPSR for CPU#2 so that the FIQ acts as a NMI
[1]	RW	0	Disable the FIQ mask bit in the CPSR for CPU#1 so that the FIQ acts as a NMI
[0]	RW	0	Disable the FIQ mask bit in the CPSR for CPU#0 so that the FIQ acts as a NMI

INT[15:0] Distributed Interrupt Controller hardware interrupts. Each bit is connected directly to the corresponding Distributed Interrupt Controller hardware interrupt line.

Dual flip flops, clocked by the output clock of the clock module, **CLKOUT**, synchronize the **nIRQ[3:0]**, **nFIQ[3:0]**, and **INT[15:0]** signals.

Interrupt Routing

A total of 32 interrupt lines, **INT[31:0]** are provided by the Distributed Interrupt Controller in the MPCore multiprocessor. Refer to the *ARM11 MPCore Processor Technical Reference Manual* (ARM DDI 0360) for further details on interrupt handling within the MPCore.

Only **INT [15:0]** are available at the input pins of the ARM11 MPCore test chip for connection to external interrupts, **INT [31:16]** are used internally. Table 4-7 lists the internal connections of **INT[31:16]** in the test chip.

Table 4-7 Internal interrupt lines

MPCore interrupt line number	Description
31	Interrupt from L220: Decode error received on master ports from L3
30	Interrupt from L220: Slave error received on master ports from L3
29	Interrupt from L220: Event counter overflow/increment
28	Interrupt from MPCore (SCU): **PMUIRQ [11]**
27	Interrupt from MPCore (SCU): **PMUIRQ [10]**
26	Interrupt from MPCore (SCU): **PMUIRQ [9]**
25	Interrupt from MPCore (SCU): **PMUIRQ [8]**
24	Interrupt from MPCore (SCU): **PMUIRQ [7]**
23	Interrupt from MPCore (SCU): **PMUIRQ [6]**
22	Interrupt from MPCore (SCU): **PMUIRQ [5]**
21	Interrupt from MPCore (SCU): **PMUIRQ [4]**
20	Interrupt from MPCore (CPU3): **PMUIRQ [3]**
19	Interrupt from MPCore (CPU2): **PMUIRQ [2]**
18	Interrupt from MPCore (CPU1): **PMUIRQ [1]**
17	Interrupt from MPCore (CPU0): **PMUIRQ [0]**
16	Tied internally to Gnd

Interrupt routing to the **INT [15:0]** input pins of the test chip is controlled by the control PLD on the CT11MP Core. Values written to bits INTMODE[1:0] in the serial write data register using the 4 wire serial interface allow the selection of three routing modes. Refer to *Interrupt routing* on page 3-18 for further details.

Debug cross-trigger matrix

The cross-trigger matrix enables use of a single debug channel that all debug acknowledge (**DBGACK[3:0]**) can feed into and all debug requests (**DBGRQ[3:0]**) can be fed from. All inputs and outputs have a corresponding enable bit in DBGMUX to remove or attach them to the channel (DBGCH).

The debug routing matrix **DBGMUX[11:0]** signals are controlled from bits SYS_PLD_CTRL1[21:10] as listed in Table 4-8:

Table 4-8 Debug matrix control bits

Debug matrix bits	Debug signals
[11]	Reserved
[10]	Reserved
[9]	Reserved
[8]	Reserved
[7]	**DBGRQ[3]**
[6]	**DBGACK[3]**
[5]	**DBGRQ[2]**
[4]	**DBGACK[2]**
[3]	**DBGRQ[1]**
[2]	**DBGACK[1]**
[1]	**DBGRQ[0]**
[0]	**DBGACK[0]**

See *SYS_PLD_CTRL1* on page 3-49 for more detail on setting the control register bits.

4.4 Power supply control

This section describes the ARM11 MPCore test chip power management.

The whole test chip has a single power domain. No *Intelligent Energy Management*, IEM or adaptive shutdown mechanism is provided. Software can put each ARM11 MPCore CPU in WFI mode. In this case, clock gating ensures that the minimum amount of logic is toggling. Some ARM11 MPCore outputs that provide power information are stored in a specific test chip register. Figure 4-8 shows the test chip power status register bit assignments.

This register is located at address:

```
Address [31:14]=PERIPHBASE [17:0]
Address [13:12]=b11, Address [11:0]=0x00C
```

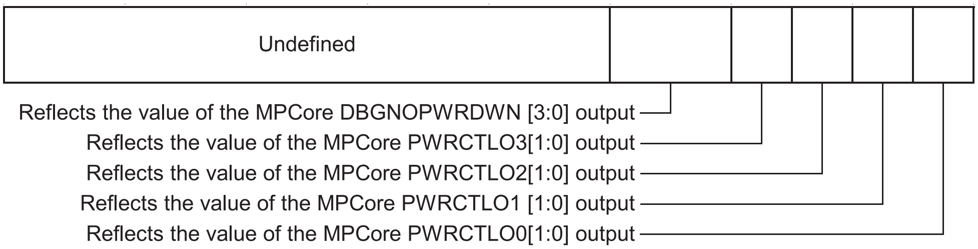

Undefined
Reflects the value of the MPCore DBGNOPWRDWN [3:0] output
Reflects the value of the MPCore PWRCTLO3[1:0] output
Reflects the value of the MPCore PWRCTLO2[1:0] output
Reflects the value of the MPCore PWRCTLO1 [1:0] output
Reflects the value of the MPCore PWRCTLO0[1:0] output

Figure 4-8 Test chip power status register

Table 4-9 lists the test chip power status register bit assignments.

Table 4-9 Test chip power status register

Bits	Read/write	Reset value	Description
[31:12]	Read as zero, write ignored.	0	Undefined.
[11:8]	RO	b0000	Reflects the value of the ARM11 MPCore **DBGNOPWRDWN [3:0]** output. See the *ARM11 MPCore Processor Technical Reference Manual* (ARM DDI 0360) for more information.
[7:6]	RO	b00	Reflects the value of the ARM11 MPCore **PWRCTLO3[1:0]** output. See the *ARM11 MPCore Processor Technical Reference Manual* (ARM DDI 0360) for more information.

Table 4-9 Test chip power status register (continued)

Bits	Read/write	Reset value	Description
[5:4]	RO	00	Reflects the value of the ARM11 MPCore **PWRCTLO2[1:0]** output. See the *ARM11 MPCore Processor Technical Reference Manual* (ARM DDI 0360) for more information.
[3:2]	RO	00	Reflects the value of the ARM11 MPCore **PWRCTLO1 1:0]** output. See the *ARM11 MPCore Processor Technical Reference Manual* (ARM DDI 0360) for more information.
[1:0]	RO	00	Reflects the value of the ARM 11 MPCore **PWRCTLO0[1:0]** output. See the *ARM11 MPCore Processor Technical Reference Manual* (ARM DDI 0360) for more information.

4.5 Memory configuration

The ARM11 MPCore test chip implements *Level 1* (L1) and *Level 2* (L2) memory subsystems.

The ARM11 MPCore L1 memory subsystem has 32KB of instruction cache and 32KB of data cache per CPU.

The L220 cache controller memory subsystem has 1MB of L2 unified cache.

——— **Note** ———

Software can change the L2 cache size using L2 control registers. See *Peripheral decoder* on page 4-29 for details of the L220 register base address allocation and the *L220 Cache Controller Technical Reference Manual* (ARM DDI 0329) for details of the L2 control registers.

4.5.1 ARM11 MPCore

The *Level 1* (L1) memory subsystem has 32KB of instruction cache and 32KB of data cache per CPU.

The ARM11 MPCore control register block requires 8KB of memory space. The **PERIPHBASE [17:0]** input, an input to the test chip, defines a space of 16KB. The ARM11 MPCore control register block uses the first 8KB.

4.5.2 L220 cache controller

The L2 memory consists of 1MB of L2 unified cache.

L2CC control register block requires 4KB memory space. The **PERIPHBASE[17:0]** input, an input to the test chip, defines a space of 16KB. The L2CC control register block is located at PERIPHBASE + 8KB.

The fixed configuration settings for the L220 in the test chip are:

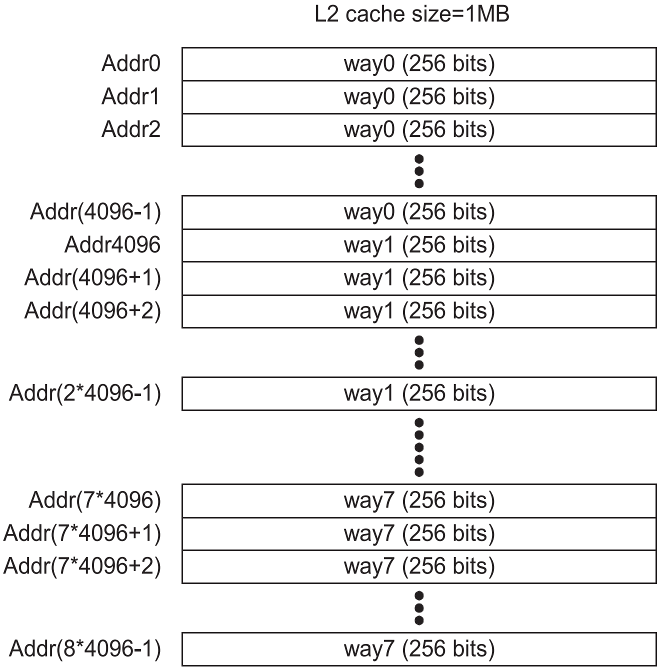
- The L220 does not use parity.
- The TrustZone feature is not required, so all transactions are considered to be secure.
- *Intelligent Energy Manager* (IEM) support is not provided.

- An external input pin, **L2MASTNUM**, enables you to select either one or two master AXI ports:
  - — **L2MASTNUM**=0 indicates one master AXI port, and master port 1 is used.
  - — **L2MASTNUM**=1 indicates two master AXI ports.
- L220 and MPCore are synchronous, and L2 cache operates at the core frequency.
- The way size is fixed at 128K.
- You can bypass the L2 cache using the L220 Control Register. This is the default state. In this case, you have an additional latency of the pipeline length of the L2 cache.

For details on the L220 Level 2 cache implementation see the *L220 Cache Controller Technical Reference Manual* (ARM DDI 0329).

RAM organization

Data RAM requires byte-enable access for data RAM without parity. Figure 4-9 on page 4-25 shows the L220 data RAM organization.



L2 cache size=1MB

Addr0	way0 (256 bits)
Addr1	way0 (256 bits)
Addr2	way0 (256 bits)
	⋮
Addr(4096-1)	way0 (256 bits)
Addr4096	way1 (256 bits)
Addr(4096+1)	way1 (256 bits)
Addr(4096+2)	way1 (256 bits)
	⋮
Addr(2*4096-1)	way1 (256 bits)
	⋮
Addr(7*4096)	way7 (256 bits)
Addr(7*4096+1)	way7 (256 bits)
Addr(7*4096+2)	way7 (256 bits)
	⋮
Addr(8*4096-1)	way7 (256 bits)

Figure 4-9 L220 data RAM organization

4.6 L220 bypass and peripheral decode

This section describes the L220 bypass mechanism and the built-in peripheral decoder. It contains the following sections:

- *L220 bypass module*
- *Peripheral decoder* on page 4-29.

4.6.1 L220 bypass module

You can use the L220 bypass module shown in Figure 4-10 on page 4-27 to bypass the L220. Whether the L220 is bypassed is determined by the **L2BYPASS** bit in the *Serial write data register*. The **L2BYPASS** bit is user configurable in the SYS_PLD_CTRL1 register implemented in the system FPGA on the EB.

It is possible to have one or two AXI ports active at the L220 and MPCore levels:

- the number of master ports you use at the L220 level is set by the **L2MASTNUM** bit in the CT11MPCore *Serial write data register*

  Note

  L2MASTNUM is not currently user configurable. For dual external AXI port operation with the EB, **L2MASTNUM** is tied HIGH by the CT11MPCore PLD making two AXI master ports active at the L220 level.

- the number of master ports you use at the MPCore level is set by the **MPMASTNUM** bit in the CT11MPCore *Serial write data register*. **MPMASTNUM** is user configurable in the SYS_PLD_CTRL1 register implemented in the EB system FPGA.

Note

In a one master configuration, the default master for the L220 is master 1. For MPCore, the default master is master 0. Therefore, inside the L220 bypass module, the two AXI bus connections are not only bypassed, but are also exchanged between ports 0 and 1.

Figure 4-10 on page 4-27 shows the L220 bypass mechanism.

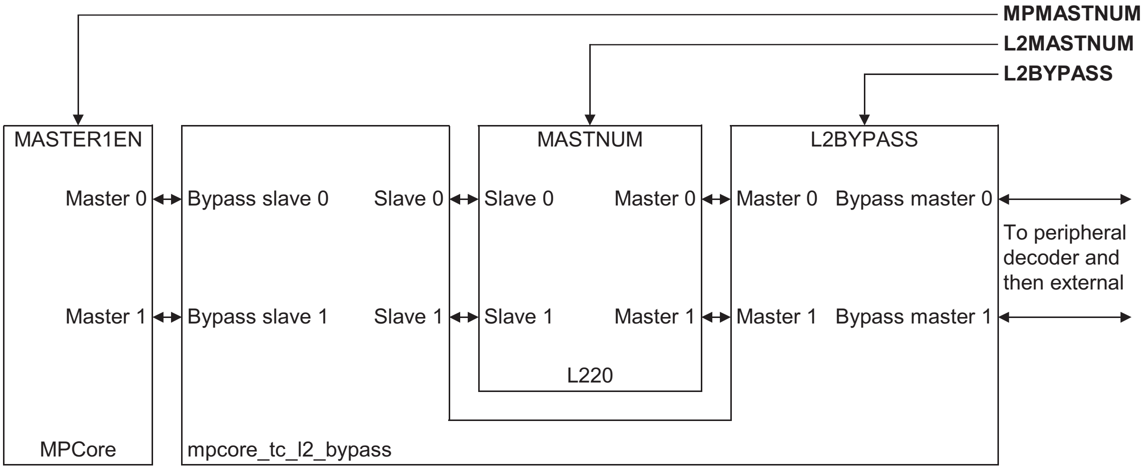


Figure 4-10 L220 bypass mechanism

Table 4-10 lists the most useful settings of the **L2BYPASS**, **MPMASTNUM**, and **L2MASTNUM** configuration bits.

Table 4-10 Most useful settings of the L2BYPASS, MPMASTNUM, and L2MASTNUM configuration bits

L2BYPASS	MPMASTNUM	L2MASTNUM	Description
0	1	0	The two MPCore masters are used. Master 1 of L220 is used, and AXI port 1 is used externally.
0	1	1	The two MPCore masters are used. Masters 0 and 1 of L220 are used, and the two AXI ports are used externally.
1	0	0	Master 0 of MPCore is used. Because AXI port 1 is used externally, the L220 bypass module exchanges the connection between port 0 and port 1.
1	1	1	Master 0 and 1 of MPCore are used, and the two AXI ports are used externally.
0	0	0	Master 0 of MPCore is used. Master 1 of L220 is used, and AXI port 1 is used externally.

Figure 4-11 on page 4-28 shows all possible ARM11 MPCore and L220 port combinations using **MPMASTNUM** and **L2MASTNUM**. The positioning of the Peripheral Decoder, is also shown. See *Peripheral decoder* on page 4-29 for further details.

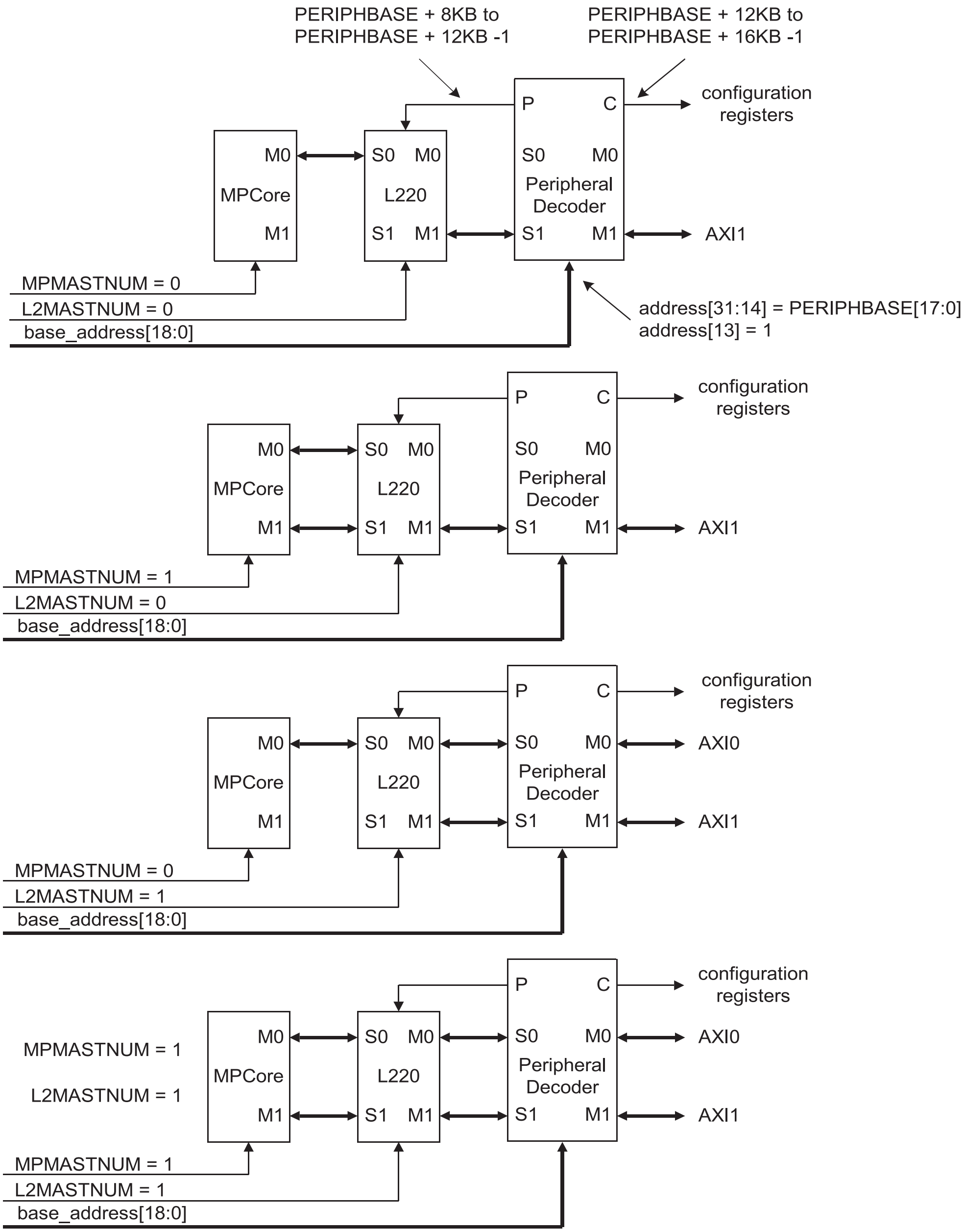


Figure 4-11 MPCore and L220 port configurations

4.6.2 Peripheral decoder

As shown in Figure 4-11 on page 4-28, the ARM11 MPCore test chip integrates some AXI decoding logic for selecting the L220 peripheral port and the test chip configuration registers. The Peripheral decoder locally decodes the regions listed in Table 4-11.

Table 4-11 Peripheral decoder mapping

Address space	Peripheral accessed
[PERIPHBASE + 8KB - PERIPHBASE + 12KB -1]	L220 control register space
[PERIPHBASE + 12KB - PERIPHBASE + 16KB -1]	Test chip-specific registers space

The PERIPHBASE address defines a 16KB memory space. You can configure this base address using an external input bus, **PERIPHBASE [17:0]**. Links are provided on the CT11MPCore to configure this bus. The ARM supplied configuration software requires the peripheral base address to be set to `0x1F000000`. This is the default link setting on the CT11MPCore.

——— **Note** ———

The ARM11 MPCore SCU decodes the region [PERIPHBASE - PERIPHBASE + 8KB -1] to control the peripheral registers of the ARM11 MPCore.

Because accesses to the peripheral ports are rare, they have priority. When a peripheral access arrives from one of the slave ports, the decoder waits for all accesses to **M0** and **M1** to complete before it acknowledges the peripheral access. This access then goes to either the **P** or the **C** port.

——— **Note** ———

During this time, no other accesses are accepted. If there are other accesses to a peripheral port from the same slave port, they are routed without wait.

The routing possibilities of the decoder are:

- idle
- normal
- **S0** to peripheral
- **S1** to peripheral.

In normal mode, accesses from **S0** are routed to **M0**, and accesses from **S1** are routed to **M1**.

——— **Note** ———

The master inputs and outputs are registered, so one cycle of delay is added for each channel.

In ‘**S0** to peripheral’ mode, all accesses from **S1** are blocked, and in ‘**S1** to peripheral’ mode, all accesses from **S0** are blocked. If an access from **S1** targets the peripheral region, it must wait for all transfers from **S0** to be completed. The state machine then goes into ‘idle’ mode, and **S1** goes into ‘peripheral’ mode.

——— **Note** ———

- The state machine always spends one cycle in ‘idle’ mode before it switches to another routing mode, irrespective of the current mode.
- Between the decoder and the downsizer in the Peripheral decoder, all the AXI channels are registered, so one delay cycle is added.

4.7 Debug and JTAG configuration

This section describes the Debug and JTAG hardware of the test chip. It contains the following sections:

- *TAP controller*
- *Debug* on page 4-32
- *JTAG Configuration* on page 4-33
- *TAP ID registers* on page 4-33.

4.7.1 TAP controller

An IEEE 1149.1 TAP controller is instantiated at the top-level of the test chip to provide a boundary scan chain. The IEEE 1149.1 TAP controller uses the same IEEE 1149.1 pins that the RealView ICE communication uses, so that the number of additional pins required is kept to a minimum.

BYPASS, EXTEST, and SAMPLE/PRELOAD instructions are the minimum functionality you require to perform the boundary scan during JTAG configuration of the test chip.

The test chip provides a specific ID code and you can obtain this using the IDCODE instruction. Table 4-12 lists the test chip TAP instructions.

Table 4-12 Test chip TAP instructions

TAP controller instruction	Instruction encoding
BYPASS	`b11111`
EXTEST	`b00000`
IDCODE	`b00001`
SAMPLE/PRELOAD	`b00010`

A separate control pin, **TESTMODE**, multiplexes the IEEE 1149.1 port. See Figure 4-12 on page 4-32.

Note

When **TESTMODE** is LOW, the TAP controllers of the ARM11 MPCore CPUs are used. A JTAG port synchronizer is also included as part of the ARM11 MPCore.

Figure 4-12 on page 4-32 shows the test chip and ARM11 MPCore TAP controller connections.

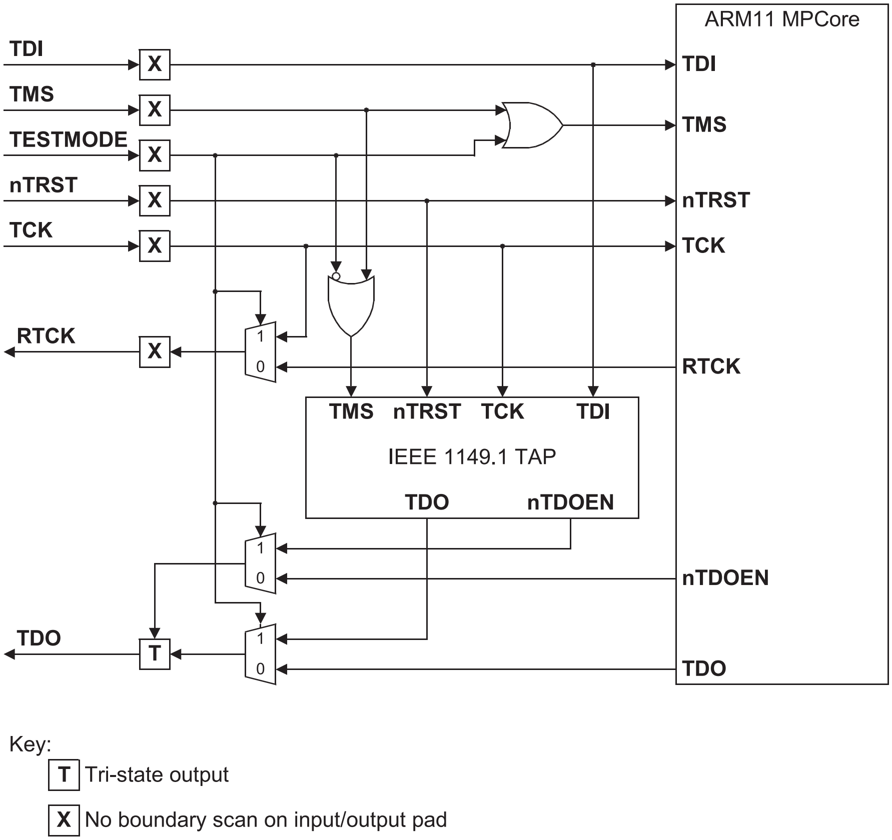


Figure 4-12 Test chip and ARM11 MPCore TAP controller connections

4.7.2 Debug

When the CONFIG slide-switch on the EB is OFF, the **TESTMODE** input is LOW and the four ARM11 MPCore DBGTAP controllers are selected, one per ARM11 MPCore CPU. You can now debug the ARM11 MPCore using a DBGTAP debugger such as RealView ICE. The four DBGTAP controllers have common **TCK**, **TMS**, and **nTRST** signals and the **TDI** and **TDO** paths are daisy-chained as shown in Figure 4-13 on page 4-33.

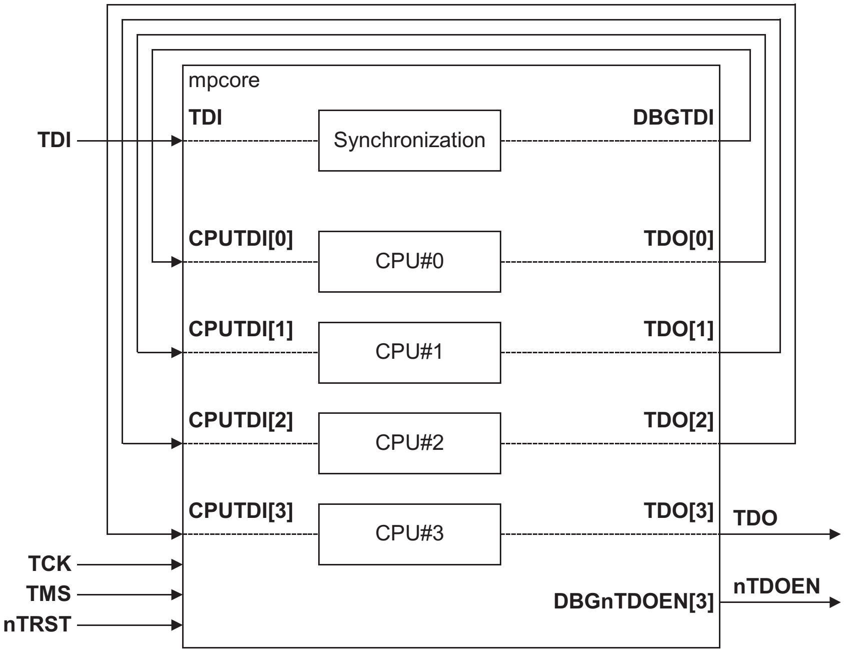


Figure 4-13 TAP signal connections between the four ARM11 MPCore CPUs

4.7.3 JTAG Configuration

When the CONFIG slide-switch on the EB is ON, the **TESTMODE** input is HIGH and the test chip boundary scan chain cells are controlled using the test chip TAP controller. You can now configure the test chip using its input and output pins.

All inputs and outputs, except for the 5-pin JTAG interface, **RTCK** and **TESTMODE**, have boundary scan cells on them. The boundary scan chain order follows that of the pads.

4.7.4 TAP ID registers

The TAP ID code is a 32-bit number divided into multiple fields. Figure 4-14 on page 4-34 shows the TAP ID register bit assignments.

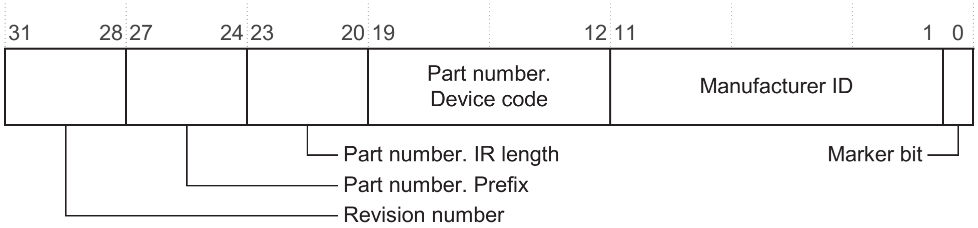


Figure 4-14 TAP ID register

You can access the *ARM11 MPCore TAP ID code* when the CT11MPCore is in Debug mode.

The ARM11 MPCore TAP ID code is:

```
0x07B37477
```

Table 4-13 lists how the register divides the number.

Table 4-13 ARM11 MPCore TAP ID register

Bit field	Use	Description	Value
[31:28]	Revision number	Revision number for the ARM11 MPCore, not necessarily the same as the test chip revision number.	`0x0`
[27:12]	Part number	[27:24] prefix. This is a fixed pattern.	`0x7`
		[23:20] IR length. Identifies the length of the instruction register within the TAP.	`0xB`
		[19:12] device code. This pattern identifies the ARM11 MPCore.	`0x37`
[11:1]	Manufacturer ID	Holds the officially registered pattern.	`b0100 0111 011`
[0]	Marker bit	Always set to 1, as required by the JTAG specification.	`b1`

See the *ARM11 MPCore Processor Technical Reference Manual* (ARM DDI 0360) for details of the remaining Debug registers in the ARM11 MPCore.

You can access the *test chip TAP ID code* when the CT11MPCore is in JTAG Configuration mode.

The test chip TAP ID is:

`0x17536021`

Table 4-13 on page 4-34 lists how the register divides the number.

Table 4-14 Test chip TAP ID register

Bit field	Use	Description	Value
[31:28]	Revision number	Revision number for the test chip, not necessarily the same as the core revision number.	`0x1`
[27:12]	Part number	[27:24] prefix. This is a fixed pattern.	`0x7`
		[23:20] IR length. Identifies the length of the instruction register within the TAP.	`0x5`
		[19:12] device code. This pattern identifies the test chip.	`0x36`
[11:1]	Manufacturer ID	Holds the officially registered pattern.	`b0000 0010 000`
[0]	Marker bit	Always set to 1, as required by the JTAG specification.	`b1`

Chapter 5
CT11MPCore Signal Descriptions

This chapter provides a summary of signals present on the CT11MPCore connectors, test points, and the links that can be modified to change signal routing. It contains the following sections:

- *Header connectors* on page 5-2
- *Links, test points, and LED indicators* on page 5-25
- *AXI bus timing specification* on page 5-33.

5.1 Header connectors

Figure 5-1 shows the pin numbers and power-blade usage of the HDRX, HDRY, and HDRZ headers on the upper side of the MPCore. For details on power supply usage see *Power supply control* on page 3-28.

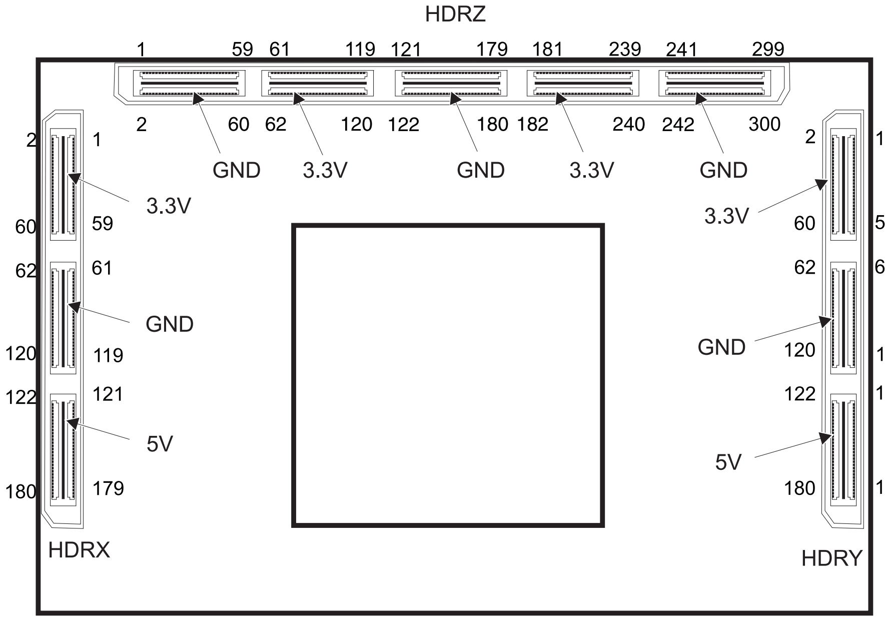


Figure 5-1 HDRX, HDRY, and HDRZ (upper) pin numbering

Table 5-1 on page 5-3 lists the Samtec part numbers.

The CT11MPCore uses the Samtec -02- connectors on the top and the Samtec -01- connectors on the bottom of the board. The total board separation is 8mm. The Core Tiles have a maximum component height of 2.5mm on the bottom and 5mm on the top of the board. This ensures that there are no component interference problems with mated boards.

Note

Samtec describes the QTH connector (used on the upper side of the tile) as a header and the QSH connector (used on the lower side of the tile) as a socket.

Table 5-1 Samtec part numbers

Header	Part number	Mating height
HDRX (upper)	QTH-090-02-F-D-A-K	8mm
HDRXL (lower)	QSH-090-01-F-D-A-K	5mm
HDRY (upper)	QTH-090-02-F-D-A-K	8mm
HDRYL (lower)	QSH-090-01-F-D-A-K	5mm
HDRZ (upper)	QTH-150-02-F-D-A-K	8mm
HDRZL (lower)	QSH-150-01-F-D-A-K	5mm

5.1.1 HDRX signals

Table 5-2 on page 5-4 describes the signals on the HDRX pins for the CT11MPCore and EB baseboard. The MPCore AXI port 1 signals are multiplexed onto the pins on this header.

——— **Note** ———

All signals on the upper and lower HDRX headers are the same for the CT11MPCore.

The signal on the upper header of the CT11MPCore is named XU*n* and the signal on the lower header is named XL*n*, the table entry is X*xn*. Replace *x* by L for the signal on the lower header and by U for the signal on the upper header. For example, the pin 1 signal for the lower header is **XL90** and the pin 1 signal for the upper header is **XU90** and the corresponding CT11MPCore signal is **ARADDR13 / ARADDR29**. This example also illustrates the designation used for a multiplexed signal, X / Y where signal X is present when **CLKOUTDIV** is HIGH and signal Y is present when **CLKOUTDIV** is LOW. See *Multiplexing scheme* on page 3-31 for further details of the AXI signal multiplexing.

For the EB there are two HDRX headers (tile site 1 and tile site 2). Replace the **TnX** signal prefix in the table by **T1X** or **T2X** to get the signal name for tile site 1 or 2 respectively.

Table 5-2 HDRX signals

Core Tile signals	Baseboard signals	Even pins	Odd pins	Baseboard signals	Core Tile signals
ARADDR12 / ARADDR28	**TnX89**	2	1	**TnX90**	**ARADDR13 / ARADDR29**
ARADDR11 / ARADDR27	**TnX88**	4	3	**TnX91**	**ARADDR14 / ARADDR30**
ARADDR10 / ARADDR26	**TnX87**	6	5	**TnX92**	**ARADDR15 / ARADDR31**
ARADDR9 / ARADDR25	**TnX86**	8	7	**TnX93**	**ARID0 / ARID2**
ARADDR8 / ARADDR24	**TnX85**	10	9	**TnX94**	**ARID1 / ARID3**
ARADDR7 / ARADDR23	**TnX84**	12	11	**TnX95**	**ARLEN0 / ARLEN2**
ARADDR6 / ARADDR22	**TnX83**	14	13	**TnX96**	**ARLEN1 / ARLEN3**
ARADDR5 / ARADDR21	**TnX82**	16	15	**TnX97**	**ARSIZE0 / ARSIZE1**
ARADDR4 / ARADDR20	**TnX81**	18	17	**TnX98**	**ARID4 / ARPROT2**
ARADDR3 / ARADDR19	**TnX80**	20	19	**TnX99**	**ARPROT0 / ARPROT1**
ARADDR2 / ARADDR18	**TnX79**	22	21	**TnX100**	**ARBURST0 / ARBURST1**
ARADDR1 / ARADDR17	**TnX78**	24	23	**TnX101**	**ARLOCK0 / ARLOCK1**
ARADDR0 / ARADDR16	**TnX77**	26	25	**TnX102**	**ARCACHE0 / ARCACHE2**
BREADY	**TnX76**	28	27	**TnX103**	**ARCACHE1 / ARCACHE3**
BVALID	**TnX75**	30	29	**TnX104**	**ARVALID /** b0

Table 5-2 HDRX signals (continued)

Core Tile signals	Baseboard signals	Even pins	Odd pins	Baseboard signals	Core Tile signals
BRESP0 / **BRESP1**	**TnX74**	32	31	**TnX105**	**ARREADY**
BID4 / (not connected)	**TnX73**	34	33	**TnX106**	**RDATA0 /** **RDATA32**
BID1 / **BID3**	**TnX72**	36	35	**TnX107**	**RDATA1 /** **RDATA33**
BID0 / **BID2**	**TnX71**	38	37	**TnX108**	**RDATA2 /** **RDATA34**
AWREADY	**TnX70**	40	39	**TnX109**	**RDATA3 /** **RDATA35**
AWVALID / b0	**TnX69**	42	41	**TnX110**	**RDATA4 /** **RDATA36**
AWCACHE1 / **AWCACHE3**	**TnX68**	44	43	**TnX111**	**RDATA5 /** **RDATA37**
AWCACHE0 / **AWCACHE2**	**TnX67**	46	45	**TnX112**	**RDATA6 /** **RDATA38**
AWLOCK0 / **AWLOCK1**	**TnX66**	48	47	**TnX113**	**RDATA7 /** **RDATA39**
AWBURST0 / **AWBURST1**	**TnX65**	50	49	**TnX114**	**RDATA8 /** **RDATA40**
AWPROT0 / **AWPROT1**	**TnX64**	52	51	**TnX115**	**RDATA9 /** **RDATA41**
ARM_nRESET	**TnX63**	54	53	**TnX116**	**RDATA10 /** **RDATA42**
AWID4 / **AWPROT2**	**TnX62**	56	55	**TnX117**	**RDATA11 /** **RDATA43**
AWSIZE0 / **AWSIZE1**	**TnX61**	58	57	**TnX118**	**RDATA12 /** **RDATA44**
AWLEN1 / **AWLEN3**	**TnX60**	60	59	**TnX119**	**RDATA13 /** **RDATA45**

Table 5-2 HDRX signals (continued)

Core Tile signals	Baseboard signals	Even pins	Odd pins	Baseboard signals	Core Tile signals
AWLEN0 / AWLEN2	**TnX59**	62	61	**TnX120**	**RDATA14 / RDATA46**
AWID1 / AWID3	**TnX58**	64	63	**TnX121**	**RDATA15 / RDATA47**
AWID0 / AWID2	**TnX57**	66	65	**TnX122**	**RDATA16 / RDATA48**
AWADDR15 / AWADDR31	**TnX56**	68	67	**TnX123**	**RDATA17 / RDATA49**
AWADDR14 / AWADDR30	**TnX55**	70	69	**TnX124**	**RDATA18 / RDATA50**
AWADDR13 / AWADDR29	**TnX54**	72	71	**TnX125**	**RDATA19 / RDATA51**
AWADDR12 / AWADDR28	**TnX53**	74	73	**TnX126**	**RDATA20 / RDATA52**
AWADDR11 / AWADDR27	**TnX52**	76	75	**TnX127**	**RDATA21 / RDATA53**
AWADDR10 / AWADDR26	**TnX51**	78	77	**TnX128**	**RDATA22 / RDATA54**
AWADDR9 / AWADDR25	**TnX50**	80	79	**TnX129**	**RDATA23 / RDATA55**
AWADDR8 / AWADDR24	**TnX49**	82	81	**TnX130**	**RDATA24 / RDATA56**
AWADDR7 / AWADDR23	**TnX48**	84	83	**TnX131**	**RDATA25 / RDATA57**
AWADDR6 / AWADDR22	**TnX47**	86	85	**TnX132**	**RDATA26 / RDATA58**
AWADDR5 / AWADDR21	**TnX46**	88	87	**TnX133**	**RDATA27 / RDATA59**
AWADDR4 / AWADDR20	**TnX45**	90	89	**TnX134**	**RDATA28 / RDATA60**

Table 5-2 HDRX signals (continued)

Core Tile signals	Baseboard signals	Even pins	Odd pins	Baseboard signals	Core Tile signals
AWADDR3 / AWADDR19	**TnX44**	92	91	**TnX135**	**RDATA29 / RDATA61**
AWADDR2 / AWADDR18	**TnX43**	94	93	**TnX136**	**RDATA30 / RDATA62**
AWADDR1 / AWADDR17	**TnX42**	96	95	**TnX137**	**RDATA31 / RDATA63**
AWADDR0 / AWADDR16	**TnX41**	98	97	**TnX138**	**RID0 / RID2**
WREADY	**TnX40**	100	99	**TnX139**	**RID1 / RID3**
WVALID / b0	**TnX39**	102	101	**TnX140**	**RRESP0 / RRESP1**
WLAST / WID4	**TnX38**	104	103	**TnX141**	**RLAST / RID4**
WSTRB3 / WSTRB7	**TnX37**	106	105	**TnX142**	**RVALID /** (not connected)
WSTRB2 / WSTRB6	**TnX36**	108	107	**TnX143**	**RREADY**
WSTRB1 / WSTRB5	**TnX35**	110	109	**TnX144**	**Xx144**
WSTRB0 / WSTRB4	**TnX34**	112	111	**TnX145**	**Xx145**
WID1 / WID3	**TnX33**	114	113	**TnX146**	**Xx146**
WID0 / WID2	**TnX32**	116	115	**TnX147**	**Xx147**
WDATA31 / WDATA63	**TnX31**	118	117	**TnX148**	**Xx148**
WDATA30 / WDATA62	**TnX30**	120	119	**TnX149**	**Xx149**

Table 5-2 HDRX signals (continued)

Core Tile signals	Baseboard signals	Even pins	Odd pins	Baseboard signals	Core Tile signals
WDATA29 / WDATA61	**TnX29**	122	121	**TnX150**	**Xx150**
WDATA28 / WDATA60	**TnX28**	124	123	**TnX151**	**Xx151**
WDATA27 / WDATA59	**TnX27**	126	125	**TnX152**	**Xx152**
WDATA26 / WDATA58	**TnX26**	128	127	**TnX153**	**Xx153**
WDATA25 / WDATA57	**TnX25**	130	129	**TnX154**	**Xx154**
WDATA24 / WDATA56	**TnX24**	132	131	**TnX155**	**Xx155**
WDATA23 / WDATA55	**TnX23**	134	133	**TnX156**	**Xx156**
WDATA22 / WDATA54	**TnX22**	136	135	**TnX157**	**Xx157**
WDATA21 / WDATA53	**TnX21**	138	137	**TnX158**	**Xx158**
WDATA20 / WDATA52	**TnX20**	140	139	**TnX159**	**Xx159**
WDATA19 / WDATA51	**TnX19**	142	141	**TnX160**	**Xx160**
WDATA18 / WDATA50	**TnX18**	144	143	**TnX161**	**Xx161**
WDATA17 / WDATA49	**TnX17**	146	145	**TnX162**	**Xx162**
WDATA16 / WDATA48	**TnX16**	148	147	**TnX163**	**Xx163**
WDATA15 / WDATA47	**TnX15**	150	149	**TnX164**	**Xx164**

Table 5-2 HDRX signals (continued)

Core Tile signals	Baseboard signals	Even pins	Odd pins	Baseboard signals	Core Tile signals
WDATA14 / WDATA46	**TnX14**	152	151	**TnX165**	**Xx165**
WDATA13 / WDATA45	**TnX13**	154	153	**TnX166**	**Xx166**
WDATA12 / WDATA44	**TnX12**	156	155	**TnX167**	**Xx167**
WDATA11 / WDATA43	**TnX11**	158	157	**TnX168**	**Xx168**
WDATA10 / WDATA42	**TnX10**	160	159	**TnX169**	**Xx169**
WDATA9 / WDATA41	**TnX9**	162	161	**TnX170**	**Xx170**
WDATA8 / WDATA40	**TnX8**	164	163	**TnX171**	**Xx171**
WDATA7 / WDATA39	**TnX7**	166	165	**TnX172**	**Xx172**
WDATA6 / WDATA38	**TnX6**	168	167	**TnX173**	**Xx173**
WDATA5 / WDATA37	**TnX5**	170	169	**TnX174**	**Xx174**
WDATA4 / WDATA36	**TnX4**	172	171	**TnX175**	**Xx175**
WDATA3 / WDATA35	**TnX3**	174	173	**TnX176**	**Xx176**
WDATA2 / WDATA34	**TnX2**	176	175	**TnX177**	**Xx177**
WDATA1 / WDATA33	**TnX1**	178	177	**TnX178**	**Xx178**
WDATA0 / WDATA32	**TnX0**	180	179	**TnX179**	**Xx179**

For a description of the AXI bus signals see *AMBA AXI Protocol Specification* (ARM IHI 0022), *Chapter 2 Signal Descriptions*.

5.1.2 HDRY signals

Table 5-3 describes the signals on the HDRY pins for the CT11MPCore and EB baseboard. The MPCore AXI port 0 signals are multiplexed onto the pins on this header.

Note

All signals on the upper and lower HDRY headers are the same for the CT11MPCore.

The signal on the upper header of the CT11MPCore is named YU*n* and the signal on the lower header is named YL*n*, the table entry is Y*xn*. Replace *x* by L for the signal on the lower header and by U for the signal on the upper header. For example, the pin 1 signal for the lower header is **YL89** and the pin 1 signal for the upper header is **YU89** and the corresponding CT11MPCore signal is **ARADDR12 / ARADDR28**. This example also illustrates the designation used for a multiplexed signal, X / Y where signal X is present when **CLKOUTDIV** is HIGH and signal Y is present when **CLKOUTDIV** is LOW. See *Multiplexing scheme* on page 3-31 for further details of the AXI signal multiplexing scheme.

For the EB there are two HDRY headers (tile site 1 and tile site 2). Replace the **TnY** signal prefix in the table by **T1Y** or **T2Y** to get the signal name for tile site 1 or 2 respectively.

Table 5-3 HDRY signals

Core Tile signals	Baseboard signals	Even pins	Odd pins	Baseboard signals	Core Tile signals
ARADDR13 / ARADDR29	**TnY90**	2	1	**TnY89**	**ARADDR12 / ARADDR28**
ARADDR14 / ARADDR30	**TnY91**	4	3	**TnY88**	**ARADDR11 / ARADDR27**
ARADDR15 / ARADDR31	**TnY92**	6	5	**TnY87**	**ARADDR10 / ARADDR26**
ARID0 / ARID2	**TnY93**	8	7	**TnY86**	**ARADDR9 / ARADDR25**
ARID1 / ARID3	**TnY94**	10	9	**TnY85**	**ARADDR8 / ARADDR24**

Table 5-3 HDRY signals (continued)

Core Tile signals	Baseboard signals	Even pins	Odd pins	Baseboard signals	Core Tile signals
ARLEN0 / **ARLEN2**	**TnY95**	12	11	**TnY84**	**ARADDR7 /** **ARADDR23**
ARLEN1 / **ARLEN3**	**TnY96**	14	13	**TnY83**	**ARADDR6 /** **ARADDR22**
ARSIZE0 / **ARSIZE1**	**TnY97**	16	15	**TnY82**	**ARADDR5 /** **ARADDR21**
ARID4 / **ARPROT2**	**TnY98**	18	17	**TnY81**	**ARADDR4 /** **ARADDR20**
ARPROT0 / **ARPROT1**	**TnY99**	20	19	**TnY80**	**ARADDR3 /** **ARADDR19**
ARBURST0 / **ARBURST1**	**TnY100**	22	21	**TnY79**	**ARADDR2 /** **ARADDR18**
ARLOCK0 / **ARLOCK1**	**TnY101**	24	23	**TnY78**	**ARADDR1 /** **ARADDR17**
ARCACHE0 / **ARCACHE2**	**TnY102**	26	25	**TnY77**	**ARADDR0 /** **ARADDR16**
ARCACHE1 / **ARCACHE3**	**TnY103**	28	27	**TnY76**	**BREADY**
ARVALID / b0	**TnY104**	30	29	**TnY75**	**BVALID**
ARREADY	**TnY105**	32	31	**TnY74**	**BRESP0 /** **BRESP1**
RDATA0 / **RDATA32**	**TnY106**	34	33	**TnY73**	**BID4 /** (not connected)
RDATA1 / **RDATA33**	**TnY107**	36	35	**TnY72**	**BID1 /** **BID3**
RDATA2 / **RDATA34**	**TnY108**	38	37	**TnY71**	**BID0 /** **BID2**
RDATA3 / **RDATA35**	**TnY109**	40	39	**TnY70**	**AWREADY**

Table 5-3 HDRY signals (continued)

Core Tile signals	Baseboard signals	Even pins	Odd pins	Baseboard signals	Core Tile signals
RDATA4 / RDATA36	**TnY110**	42	41	**TnY69**	**AWVALID /** b0
RDATA5 / RDATA37	**TnY111**	44	43	**TnY68**	**AWCACHE1 / AWCACHE3**
RDATA6 / RDATA38	**TnY112**	46	45	**TnY67**	**AWCACHE0 / AWCACHE2**
RDATA7 / RDATA39	**TnY113**	48	47	**TnY66**	**AWLOCK0 / AWLOCK1**
RDATA8 / RDATA40	**TnY114**	50	49	**TnY65**	**AWBURST0 / AWBURST1**
RDATA9 / RDATA41	**TnY115**	52	51	**TnY64**	**AWPROT0 / AWPROT1**
RDATA10 / RDATA42	**TnY116**	54	53	**TnY63**	**ARM_nRESET**
RDATA11 / RDATA43	**TnY117**	56	55	**TnY62**	**AWID4 / AWPROT2**
RDATA12 / RDATA44	**TnY118**	58	57	**TnY61**	**AWSIZE0 / AWSIZE1**
RDATA13 / RDATA45	**TnY119**	60	59	**TnY60**	**AWLEN1 / AWLEN3**
RDATA14 / RDATA46	**TnY120**	62	61	**TnY59**	**AWLEN0 / AWLEN2**
RDATA15 / RDATA47	**TnY121**	64	63	**TnY58**	**AWID1 / AWID3**
RDATA16 / RDATA48	**TnY122**	66	65	**TnY57**	**AWID0 / AWID2**
RDATA17 / RDATA49	**TnY123**	68	67	**TnY56**	**AWADDR15 / AWADDR31**
RDATA18 / RDATA50	**TnY124**	70	69	**TnY55**	**AWADDR14 / AWADDR30**

Table 5-3 HDRY signals (continued)

Core Tile signals	Baseboard signals	Even pins	Odd pins	Baseboard signals	Core Tile signals
RDATA19 / RDATA51	**TnY125**	72	71	**TnY54**	**AWADDR13 / AWADDR29**
RDATA20 / RDATA52	**TnY126**	74	73	**TnY53**	**AWADDR12 / AWADDR28**
RDATA21 / RDATA53	**TnY127**	76	75	**TnY52**	**AWADDR11 / AWADDR27**
RDATA22 / RDATA54	**TnY128**	78	77	**TnY51**	**AWADDR10 / AWADDR26**
RDATA23 / RDATA55	**TnY129**	80	79	**TnY50**	**AWADDR9 / AWADDR25**
RDATA24 / RDATA56	**TnY130**	82	81	**TnY49**	**AWADDR8 / AWADDR24**
RDATA25 / RDATA57	**TnY131**	84	83	**TnY48**	**AWADDR7 / AWADDR23**
RDATA26 / RDATA58	**TnY132**	86	85	**TnY47**	**AWADDR6 / AWADDR22**
RDATA27 / RDATA59	**TnY133**	88	87	**TnY46**	**AWADDR5 / AWADDR21**
RDATA28 / RDATA60	**TnY134**	90	89	**TnY45**	**AWADDR4 / AWADDR20**
RDATA29 / RDATA61	**TnY135**	92	91	**TnY44**	**AWADDR3 / AWADDR19**
RDATA30 / RDATA62	**TnY136**	94	93	**TnY43**	**AWADDR2 / AWADDR18**
RDATA31 / RDATA63	**TnY137**	96	95	**TnY42**	**AWADDR1 / AWADDR17**
RID0 / RID2	**TnY138**	98	97	**TnY41**	**AWADDR0 / AWADDR16**
RID1 / RID3	**TnY139**	100	99	**TnY40**	**WREADY**

Table 5-3 HDRY signals (continued)

Core Tile signals	Baseboard signals	Even pins	Odd pins	Baseboard signals	Core Tile signals
RRESP0 / **RRESP1**	**TnY140**	102	101	**TnY39**	**WVALID /** b0
RLAST / **RID4**	**TnY141**	104	103	**TnY38**	**WLAST /** **WID4**
RVALID / (not connected)	**TnY142**	106	105	**TnY37**	**WSTRB3 /** **WSTRB7**
RREADY	**TnY143**	108	107	**TnY36**	**WSTRB2 /** **WSTRB6**
Yx144	**TnY144**	110	109	**TnY35**	**WSTRB1 /** **WSTRB5**
Yx145	**TnY145**	112	111	**TnY34**	**WSTRB0 /** **WSTRB4**
Yx146	**TnY146**	114	113	**TnY33**	**WID1 /** **WID3**
Yx147	**TnY147**	116	115	**TnY32**	**WID0 /** **WID2**
Yx148	**TnY148**	118	117	**TnY31**	**WDATA31 /** **WDATA63**
Yx149	**TnY149**	120	119	**TnY30**	**WDATA30 /** **WDATA62**
Yx150	**TnY150**	122	121	**TnY29**	**WDATA29 /** **WDATA61**
Yx151	**TnY151**	124	123	**TnY28**	**WDATA28 /** **WDATA60**
Yx152	**TnY152**	126	125	**TnY27**	**WDATA27 /** **WDATA59**
Yx153	**TnY153**	128	127	**TnY26**	**WDATA26 /** **WDATA58**
Yx154	**TnY154**	130	129	**TnY25**	**WDATA25 /** **WDATA57**

Table 5-3 HDRY signals (continued)

Core Tile signals	Baseboard signals	Even pins	Odd pins	Baseboard signals	Core Tile signals
Yx155	**TnY155**	132	131	**TnY24**	**WDATA24 / WDATA56**
Yx156	**TnY156**	134	133	**TnY23**	**WDATA23 / WDATA55**
Yx157	**TnY157**	136	135	**TnY22**	**WDATA22 / WDATA54**
Yx158	**TnY158**	138	137	**TnY21**	**WDATA21 / WDATA53**
Yx159	**TnY159**	140	139	**TnY20**	**WDATA20 / WDATA52**
Yx160	**TnY160**	142	141	**TnY19**	**WDATA19 / WDATA51**
Yx161	**TnY161**	144	143	**TnY18**	**WDATA18 / WDATA50**
Yx162	**TnY162**	146	145	**TnY17**	**WDATA17 / WDATA49**
Yx163	**TnY163**	148	147	**TnY16**	**WDATA16 / WDATA48**
Yx164	**TnY164**	150	149	**TnY15**	**WDATA15 / WDATA47**
Yx165	**TnY165**	152	151	**TnY14**	**WDATA14 / WDATA46**
Yx166	**TnY166**	154	153	**TnY13**	**WDATA13 / WDATA45**
Yx167	**TnY167**	156	155	**TnY12**	**WDATA12 / WDATA44**
Yx168	**TnY168**	158	157	**TnY11**	**WDATA11 / WDATA43**
Yx169	**TnY169**	160	159	**TnY10**	**WDATA10 / WDATA42**

Table 5-3 HDRY signals (continued)

Core Tile signals	Baseboard signals	Even pins	Odd pins	Baseboard signals	Core Tile signals
Yx160	**TnY170**	162	161	**TnY9**	**WDATA9 / WDATA41**
Yx171	**TnY171**	164	163	**TnY8**	**WDATA8 / WDATA40**
Yx172	**TnY172**	166	165	**TnY7**	**WDATA7 / WDATA39**
Yx173	**TnY173**	168	167	**TnY6**	**WDATA6 / WDATA38**
Yx174	**TnY174**	170	169	**TnY5**	**WDATA5 / WDATA37**
Yx175	**TnY175**	172	171	**TnY4**	**WDATA4 / WDATA36**
Yx176	**TnY176**	174	173	**TnY3**	**WDATA3 / WDATA35**
Yx177	**TnY177**	176	175	**TnY2**	**WDATA2 / WDATA34**
Yx178	**TnY178**	178	177	**TnY1**	**WDATA1 / WDATA33**
Yx179	**TnY179**	180	179	**TnY0**	**WDATA0 / WDATA32**

For a description of the AXI bus signals see *AMBA AXI Protocol Specification* (ARM IHI 0022), *Chapter 2 Signal Descriptions*.

5.1.3 HDRZ signals

Table 5-4 on page 5-17 describes the signals on the HDRZ pins for the CT11MPCore and the EB.

——— **Note** ————

Except for the clock and JTAG signals, and **nTILE_DET**, the CT11MCore either has the same signal on both the upper and lower pins or a signal on the lower pins only.

Some of the HDRZ signals can be isolated from the HDRZ header pins and these are shown bracketed in the table listing. See *HDRZ signal mux connections* on page 3-25 for details.

Some of the HDRZ signals are multiplexed and use the designation `X / Y` where signal `X` is present when **CLKOUTDIV** is HIGH and signal `Y` is present when **CLKOUTDIV** is LOW. See *Multiplexing scheme* on page 3-31 for further details of the AXI signal multiplexing scheme.

The signal on the upper header of the CT11MPCore is named `ZU`*n* and the signal on the lower header is named `ZL`*n*, the table entry is `Z`*xn*. Replace *x* by `L` for the signal on the lower header and by `U` for the signal on the upper header. For example, the pin 1 signal for the lower header is **ZL128** and the pin 1 signal for the upper header is **ZU128**.

For the EB there are two HDRZ headers (tile site 1 and tile site 2). Replace the **TnZ** signal prefix in the table by **T1Z** or **T2Z** to get the signal name for tile site 1 or 2 re respectively.

Table 5-4 HDRZ signals

Core Tile signals	Baseboard signals	Even pins	Odd pins	Baseboard signals	Core Tile signals
Z255	**TnZ255**	2	1	**TnZ128**	**Zx128**
Z254	**TnZ254**	4	3	**TnZ129**	**Zx129**
Z253	**TnZ253**	6	5	**TnZ130**	**Zx130**
Z252	**TnZ252**	8	7	**TnZ131**	**Zx131**
Z251	**TnZ251**	10	9	**TnZ132**	**Zx132**
Z250	**TnZ250**	12	11	**TnZ133**	**Zx133**
Z249	**TnZ249**	14	13	**TnZ134**	**Zx134**
Z248	**TnZ248**	16	15	**TnZ135**	**Zx135**
Z247	**TnZ247**	18	17	**TnZ136**	**Zx136**
Z246	**TnZ246**	20	19	**TnZ137**	**Zx137**
Z245	**TnZ245**	22	21	**TnZ138**	**Zx138**
Z244	**TnZ244**	24	23	**TnZ139**	**Zx139**

Table 5-4 HDRZ signals (continued)

Core Tile signals	Baseboard signals	Even pins	Odd pins	Baseboard signals	Core Tile signals
Z243	**TnZ243**	26	25	**TnZ140**	**Zx140**
Z242	**TnZ242**	28	27	**TnZ141**	**Zx141**
Z241	**TnZ241**	30	29	**TnZ142**	**Zx142**
Z240	**TnZ240**	32	31	**TnZ143**	**Zx143**
Z239	**TnZ249**	34	33	**TnZ144**	**Zx144**
Z238	**TnZ248**	36	35	**TnZ145**	**Zx145**
Z237	**TnZ237**	38	37	**TnZ146**	**Zx146**
Z236	**TnZ236**	40	39	**TnZ147**	**Zx147**
Z235	**TnZ235**	42	41	**TnZ148**	**Zx148**
Z234	**TnZ234**	44	43	**TnZ149**	**Zx149**
Z233	**TnZ233**	46	45	**TnZ150**	**Zx150**
Z232	**TnZ232**	48	47	**TnZ151**	**Zx151**
nWARMRST	**TnZ231**	50	49	**TnZ152**	**Zx152**
PLDCLK	**TnZ230**	52	51	**TnZ153**	**Zx153**
PLDnRESET	**TnZ229**	54	53	**TnZ154**	**Zx154**
PLDD0	**TnZ228**	56	55	**TnZ155**	**Zx155**
PLDD1	**TnZ227**	58	57	**TnZ156**	**Zx156**
AWUSER1_SW4 / ARUSER1_SW4	**TnZ226**	60	59	**TnZ157**	**Zx157**
AWUSER1_SW3 / ARUSER1_SW3	**TnZ225**	62	61	**TnZ158**	**Zx158**
AWUSER1_SW2 / ARUSER1_SW2	**TnZ224**	64	63	**TnZ159**	**Zx159**
AWUSER1_SW1 / ARUSER1_SW1	**TnZ223**	66	65	**TnZ160**	**Zx160**
AWUSER1_SW0 / ARUSER1_SW0	**TnZ222**	68	67	**TnZ161**	**Zx161**

Table 5-4 HDRZ signals (continued)

Core Tile signals	Baseboard signals	Even pins	Odd pins	Baseboard signals	Core Tile signals
AWUSER0_SW4 / ARUSER0_SW4	**TnZ221**	70	69	**TnZ162**	**Zx162**
AWUSER0_SW3 / ARUSER0_SW3	**TnZ220**	72	71	**TnZ163**	**Zx163**
AWUSER0_SW2 / ARUSER0_SW2	**TnZ219**	74	73	**TnZ164**	**Zx164**
AWUSER0_SW1 / ARUSER0_SW1	**Zx218**	76	75	**TnZ165**	**Zx165**
AWUSER0_SW0 / ARUSER0_SW0	**Zx217**	78	77	**TnZ166**	**Zx166**
nCOLDRST	**Zx216**	80	79	**TnZ167**	**Zx167**
INT15	**Zx215**	82	81	**TnZ168**	**Zx168 (nPORESET0)**
INT14	**Zx214**	84	83	**TnZ169**	**Zx169 (nPORESET1)**
INT13	**Zx213**	86	85	**TnZ170**	**Zx170 (nPORESET2)**
INT12	**Zx212**	88	87	**TnZ171**	**Zx171 (nPORESET3)**
INT11	**Zx211**	90	89	**TnZ172**	**Zx172 (nCPURESET0)**
INT10	**Zx210**	92	91	**TnZ173**	**Zx173 (nCPURESET1)**
INT9	**Zx209**	94	93	**TnZ174**	**Zx174 (nCPURESET2)**
INT8	**Zx208**	96	95	**TnZ175**	**Zx175 (nCPURESET3)**
INT7	**Zx207**	98	97	**TnZ176**	**Zx176 (RESETREQ0)**

Table 5-4 HDRZ signals (continued)

Core Tile signals	Baseboard signals	Even pins	Odd pins	Baseboard signals	Core Tile signals
INT6	**Zx206**	100	99	**TnZ177**	**Zx177** (**RESETREQ1**)
INT5	**Zx205**	102	101	**TnZ178**	**Zx178** (**RESETREQ2**)
INT4	**Zx204**	104	103	**TnZ179**	**Zx179** (**RESETREQ3**)
INT3	**Zx203**	106	105	**TnZ180**	**Zx180** (**nWDRESET0**)
INT2	**Zx202**	108	107	**TnZ181**	**Zx181** (**nWDRESET1**)
INT1	**Zx201**	110	109	**TnZ182**	**Zx182** (**nWDRESET2**)
INT0	**Zx200**	112	111	**TnZ183**	**Zx183** (**nWDRESET3**)
Zx199 (**STNBYWFI3**)	**Zx199**	114	113	**TnZ184**	**Zx184** (**EDBGRQ0**)
Zx198 (**STNBYWFI2**)	**Zx198**	116	115	**TnZ185**	**Zx185** (**EDBGRQ1**)
Zx197 (**STNBYWFI1**)	**Zx197**	118	117	**TnZ186**	**Zx186** (**EDBGRQ2**)
Zx196 (**STNBYWFI0**)	**Zx196**	120	119	**TnZ187**	**Zx187** (**EDBGRQ3**)
Zx195 (**SMPnAMP3**)	**Zx195**	122	121	**TnZ188**	**Zx188** (**DBGACK0**)
Zx194 (**SMPnAMP2**)	**Zx194**	124	123	**TnZ189**	**Zx189** (**DBGACK1**)
Zx193 (**SMPnAMP1**)	**Zx193**	126	125	**TnZ190**	**Zx190** (**DBGACK2**)
Zx192 (**SMPnAMP0**)	**Zx192**	128	127	**TnZ191**	**Zx191** (**DBGACK3**)

Table 5-4 HDRZ signals (continued)

Core Tile signals	Baseboard signals	Even pins	Odd pins	Baseboard signals	Core Tile signals
CLK_POS_DN	**CLK_POS_DN_IN**	130	129	**D_nSRST**	**D_nSRST**
CLK_NEG_DN_IN **CLK_NEG_DN_OUT**	**CLK_NEG_DN_IN**	132 (U) 132 (L)	131	**D_nTRST**	**D_nTRST**
CLK_POS_UP	**CLK_POS_UP_OUT**	134	133 (U) 133 (L)	**D_TDO_IN**	**D_TDO_IN** **D_TDO_OUT**
CLK_NEG_UP_OUT **CLK_NEG_UP_IN**	**CLK_NEG_UP_OUT**	136 (U) 136 (L)	135	**D_TDI**	**D_TDI**
CLK_UP_MINUS2	**CLK_UP_THRU**	138	137 (U) 137 (L)	**D_TCK_OUT**	**D_TCK_OUT** **D_TCK_IN**
CLK_UP_MINUS1	**CLK_OUT_PLUS1**	140	139 (U) 139 (L)	**D_TMS_OUT**	**D_TMS_OUT** **D_TMS_IN**
CLK_UP_THRU	**CLK_OUT_PLUS2**	142	141	**D_RTCK**	**D_RTCK**
CLK_DN_PLUS2	**CLK_IN_PLUS2**	144	143	**C_nSRST**	**C_nSRST**
CLK_DN_PLUS1	**CLK_IN_PLUS1**	146	145	**C_nTRST**	**C_nTRST**
CLK_DN_THRU	**CLK_DN_THRU**	148	147 (U) 147 (L)	**C_TDO_IN**	**C_TDO_IN** **C_TDO_OUT**
CLK_GLOBAL	**CLK_GLOBAL**	150	149	**C_TDI**	**C_TDI**
FPGA_IMAGE	**FPGA_IMAGE**	152	151 (U) 151 (L)	**C_TCK_OUT**	**C_TCK_OUT** **C_TCK_IN**
nSYSPOR	**nSYSPOR**	154	153 (U) 153 (L)	**C_TMS_OUT**	**C_TMS_OUT** **C_TMS_IN**
nSYSRST	**nSYSRST**	156	155 (U) 155 (L)	**nTILE_DET**	**nTILE_DET** **GND**
nRTCKEN	**nRTCKEN**	158	157	**nCFGEN**	**nCFGEN**
SPARE12	**SPARE12** (reserved)	160	159	**GLOBAL_DONE**	**GLOBAL_DONE**
SPARE10	**SPARE10** (reserved)	162	161	**SPARE11** (reserved)	**SPARE11**
SPARE8	**SPARE8** (reserved)	164	163	**SPARE9** (reserved)	**SPARE9**

Table 5-4 HDRZ signals (continued)

Core Tile signals	Baseboard signals	Even pins	Odd pins	Baseboard signals	Core Tile signals
SPARE6	**SPARE6** (reserved)	166	165	**SPARE7** (reserved)	**SPARE7**
SPARE4	**SPARE4** (reserved)	168	167	**SPARE5** (reserved)	**SPARE5**
SPARE2	**SPARE2** (reserved)	170	169	**SPARE3** (reserved)	**SPARE3**
SPARE0	**SPARE0** (reserved)	172	171	**SPARE1** (reserved)	**SPARE1**
Zx64	**TnZ64**	174	173	**TnZ63**	**Zx63**
Zx65	**TnZ65**	176	175	**TnZ62**	**Zx62**
Zx66	**TnZ66**	178	177	**TnZ61**	**Zx61**
Zx67	**TnZ67**	180	179	**TnZ60**	**Zx60**
Zx68	**TnZ68**	182	181	**TnZ59**	**Zx59**
Zx69	**TnZ79**	184	183	**TnZ58**	**Zx58**
Zx70	**TnZ70**	186	185	**TnZ57**	**Zx57**
Zx71	**TnZ71**	188	187	**TnZ56**	**Zx56**
Zx72	**TnZ72**	190	189	**TnZ55**	**Zx55**
Zx73	**TnZ73**	192	191	**TnZ54**	**Zx54**
Zx74	**TnZ74**	194	193	**vZ53**	**Zx53**
Zx75	**TnZ75**	196	195	**TnZ52**	**Zx52**
Zx76	**TnZ76**	198	197	**TnZ51**	**Zx51**
Zx77	**TnZ77**	200	199	**TnZ50**	**Zx50**
Zx78	**TnZ78**	202	201	**TnZ49**	**Zx49**
Zx79	**TnZ79**	204	203	**TnZ48**	**Zx48**
Zx80	**TnZ80**	206	205	**TnZ47**	**Zx47**
Zx81	**TnZ81**	208	207	**TnZ46**	**Zx46**
Zx82	**TnZ82**	210	209	**TnZ45**	**Zx45**
Zx83	**TnZ83**	212	211	**TnZ44**	**Zx44**

Table 5-4 HDRZ signals (continued)

Core Tile signals	Baseboard signals	Even pins	Odd pins	Baseboard signals	Core Tile signals
Zx84	**TnZ84**	214	213	**TnZ43**	**Zx43**
Zx85	**TnZ85**	216	215	**TnZ42**	**Zx42**
Zx86	**TnZ86**	218	217	**TnZ41**	**Zx41**
Zx87	**TnZ87**	220	219	**TnZ40**	**Zx40**
Zx88	**TnZ88**	222	221	**TnZ39**	**Zx39**
Zx89	**TnZ89**	224	223	**TnZ38**	**Zx38**
Zx90	**TnZ90**	226	225	**TnZ37**	**Zx37**
Zx91	**TnZ91**	228	227	**TnZ36**	**Zx36**
Zx92	**TnZ92**	230	229	**TnZ35**	**Zx35**
Zx93	**TnZ93**	232	231	**TnZ34**	**Zx34**
Zx94	**TnZ94**	234	233	**TnZ33**	**Zx33**
Zx95	**TnZ95**	236	235	**TnZ32**	**Zx32**
Zx96	**TnZ96**	238	237	**TnZ31**	**Zx31**
Zx97	**TnZ97**	240	239	**TnZ30**	**Zx30**
Zx98	**TnZ98**	242	241	**TnZ29**	**Zx29**
Zx99	**TnZ99**	244	243	**TnZ28**	**Zx28**
Zx100	**TnZ100**	246	245	**TnZ27**	**Zx27**
Zx101	**TnZ101**	248	247	**TnZ26**	**Zx26**
Zx102	**TnZ102**	250	249	**TnZ25**	**Zx25**
Zx103	**TnZ103**	252	251	**TnZ24**	**Zx24**
Zx104	**TnZ104**	254	253	**TnZ23**	**Zx23**
Zx105	**TnZ105**	256	255	**TnZ22**	**Zx22**
Zx106	**TnZ106**	258	257	**TnZ21**	**Zx21**
Zx107	**TnZ107**	260	259	**TnZ20**	**Zx20**

Table 5-4 HDRZ signals (continued)

Core Tile signals	Baseboard signals	Even pins	Odd pins	Baseboard signals	Core Tile signals
Zx108	**TnZ108**	262	261	**TnZ19**	**Zx19**
Zx109	**TnZ109**	264	263	**TnZ18**	**Zx18**
Zx110	**TnZ110**	266	265	**TnZ17**	**Zx17**
Zx111	**TnZ111**	268	267	**TnZ16**	**Zx16**
Zx112	**TnZ112**	270	269	**TnZ15**	**Zx15**
Zx113	**TnZ113**	272	271	**TnZ14**	**Zx14**
Zx114	**TnZ114**	274	273	**TnZ13**	**Zx13**
Zx115	**TnZ115**	276	275	**TnZ12**	**Zx12**
Zx116	**TnZ116**	278	277	**TnZ11**	**Zx11**
Zx117	**TnZ117**	280	279	**TnZ10**	**Zx10**
Zx118	**TnZ118**	282	281	**TnZ9**	**Zx9**
Zx119	**TnZ119**	284	283	**TnZ8**	**Zx8**
Zx120	**TnZ120**	286	285	**TnZ7**	**Zx7**
Zx121	**TnZ121**	288	287	**TnZ6**	**Zx6**
Zx122	**TnZ122**	290	289	**TnZ5**	**Zx5**
Zx123	**TnZ123**	292	291	**TnZ4**	**Zx4**
Zx124	**TnZ124**	294	293	**TnZ3**	**Zx3**
Zx125	**TnZ125**	296	295	**TnZ2**	**Zx2**
Zx126	**TnZ126**	298	297	**TnZ1**	**Zx1**
Zx127	**TnZ127**	300	299	**TnZ0**	**Zx0**

5.2 Links, test points, and LED indicators

This section describes the links, test points and LED indicators present on the CT11MPCore.

5.2.1 Links

Figure 5-2 shows the location of the links on the bottom of the CT11MPCore. Figure 5-3 on page 5-26 shows the location of the links on the top of the CT11MPCore. The function of each link is listed in Table 5-5 on page 5-27.

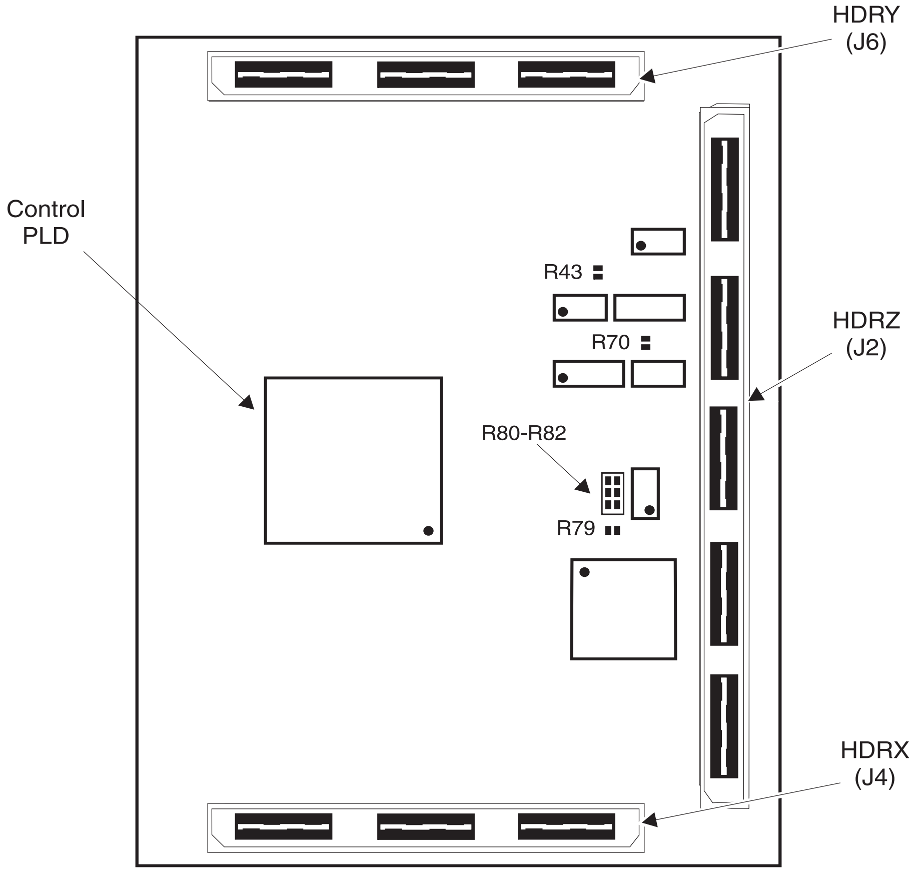


Figure 5-2 Location of links (bottom)

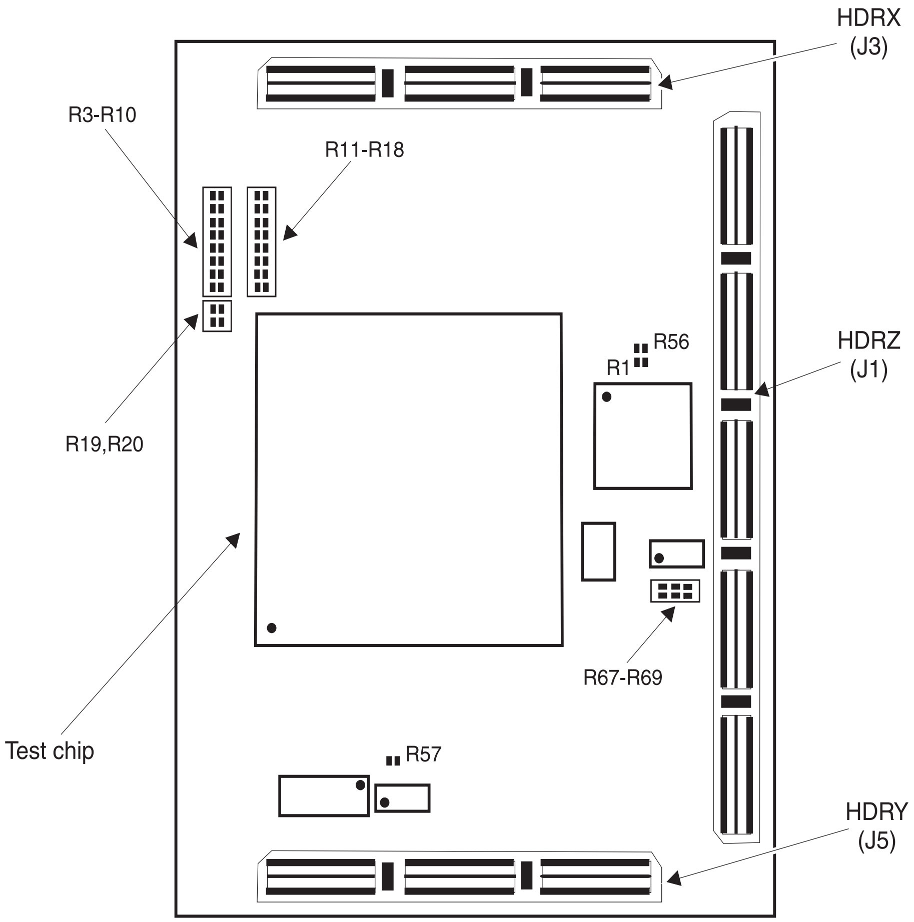


Figure 5-3 Location of links (top)

Table 5-5 Link function

Link	Description
R1	When fitted, the ISP1 clock generator output is enabled. When not fitted, the ISP1 clock generator output is either HIGH or LOW. **Note** This link is fitted by default and is required for normal operation.
R3-R20	**PERIPHBASE[17:0]** value. Defines bits [31:14] of the 32 bit peripheral address for the ARM11 MPCore test chip. Link R3 defines bit 0, link R4 defines bit 1, and so on through to R20 which defines bit 17. A bit is tied LOW, through a 10KΩ resistor, unless a link is fitted to force the bit HIGH by direct connection to the 3V3 supply. The default setting of **PERIPHBASE[17:0]** is b000111110000000000 (0x1F00 0000). See *ARM11 MPCore test chip registers* on page 3-44.
R43	When fitted, **nRTCKEN** is tied LOW in normal mode and floats in configuration mode. When not fitted, **nRTCKEN** floats unless controlled via header HDRZ. **Note** When the CT11MPCore is used with the EB this link is not fitted.

Table 5-5 Link function (continued)

Link	Description
R56	When fitted, the ISP0 clock generator output is enabled. When not fitted, the ISP0 clock generator output is either HIGH or LOW. **Note** This link is fitted by default and is required for normal operation.
R57	When fitted, enables the AXI bus switches. When not fitted, the AXI bus switches are isolated for test purposes. **Note** When the CT11MPCore is used with the EB this link is fitted to provide dual AXI ports.
R67-70, R79-R82	JTAG configuration mode routing options. The default routing option, when R79, R81, R68, and R70 are fitted, connects all JTAG configurable devices, the ISP clock generator 0, ISP clock generator 1, the PLD, and the ARM11 MPCore test chip in the configuration scan chain. **Note** This link option is the only supported option and is required for normal operation. Other available option are for production test purposes only.

5.2.2 Test points

Figure 5-4 on page 5-29 shows the location of test points on the CT11MPCore. Table 5-6 on page 5-29 lists the function of the signal at each test point.

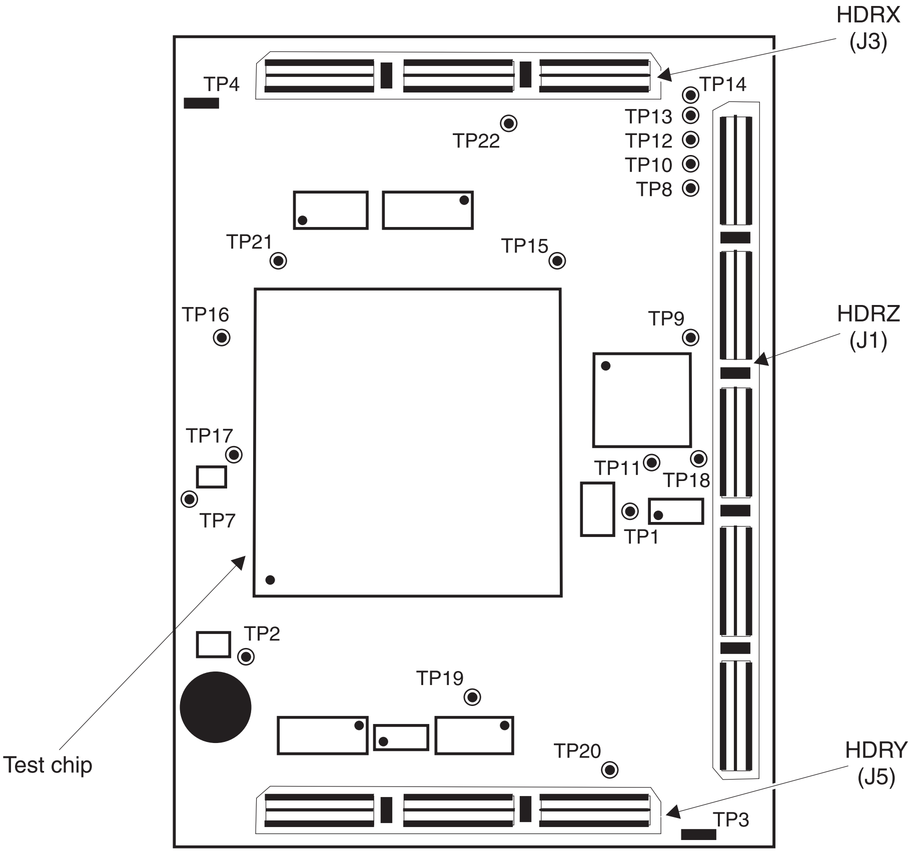


Figure 5-4 Test point location

Table 5-6 Test point signal and description

Test point	Signal	Description
TP1	**REFOUT0**	Local clock generator CLK1 output.
TP2	**VDDCORE**	1V2 supply to the ARM11 MPCore test chip core.
TP3	**GND**	Ground (0V) reference loop for signal measurements.
TP4	**GND**	Ground (0V) reference loop for signal measurements.
TP7	**PLD_SPARE0**	Function not implemented, used in development only,

Table 5-6 Test point signal and description (continued)

Test point	Signal	Description
TP8	**nLOCK1**	Lock indicator. Indicates that the ISP1 clock generator is locked to **CLKOUTDIV** from test chip PLL.
TP9	**ISP1_9A**	ISP1 clock generator output Bank 9A.
TP10	**ISP1_9B**	ISP1 clock generator output Bank 9B.
TP11	**CLKOUTDIV**	Input clock to ISP1 clock generator from the test chip PLL.
TP12	**nLOCK0**	Lock indicator. Indicates that the ISP0 clock generator is locked to **CLKOUTDIVD** from the test chip PLL.
TP13	**ISP0_7A**	ISP0 clock generator output Bank 7A.
TP14	**ISP0_7B**	ISP0 clock generator output Bank 7B.
TP15	**CLKOUTDIVD**	Input clock to ISP0 clock generator from the test chip PLL.
TP16	**AVDD**	1V2 supply to the ARM11 MPCore test chip PLL.
TP17	**1V8**	1V8 supply to the PLD core.
TP18	**REFCLK**	Reference clock from selected external source.
TP19	**RVALID0**	**RVALID** from AXI port 0.
TP20	**ARVALID0**	**ARVALID** from AXI port 0.
TP21	**RVALID1**	**RVALID** from AXI port 1.
TP22	**ARVALID1**	**ARVALID** from AXI port 1.

Note

TP5 and TP6 are not included on the production release of the Core Tile.

5.2.3 LED indicators

Figure 5-5 on page 5-31 shows the location of LED indicators on the CT11MPCore. Table 5-7 on page 5-31 lists the function of each LED indicator.

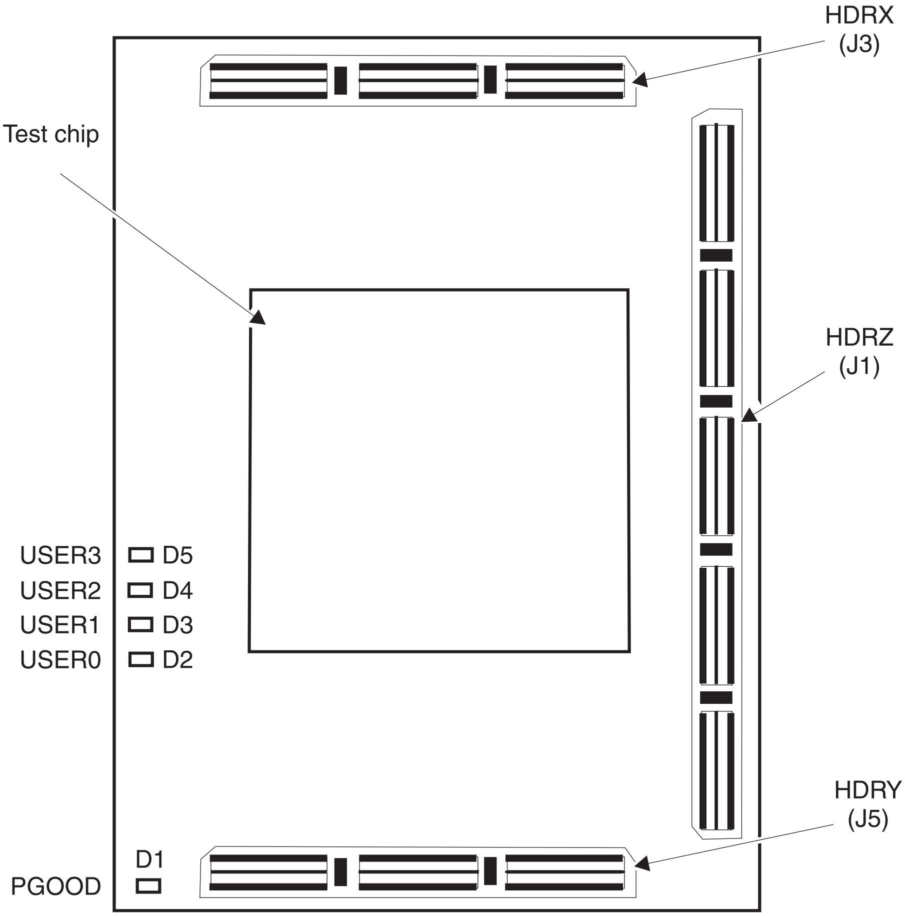


Figure 5-5 LED indicator location

Table 5-7 LED function

LED	Color	Description
PGOOD	Green	Indicates that the ARM11 MPCore test chip VDDCORE power supply is ON and within tolerance.
USER0	Green	Indicates by 1Hz flashing that the serial link to the CT11MPCore control PLD is locked.

Table 5-7 LED function (continued)

LED	Color	Description
USER1	Green	Indicates that the ISP1 clock generator is locked to the **CLKOUTDIV** PLL output clock.
USER2	Green	Indicates that the ISP0 clock generator is locked to the **CLKOUTDIVD** PLL output clock.
USER3	Green	Indicates that the ARM11 MPCore test chip PLL is locked to the **REFCLK** input clock.

Note

The USER[3:0] LED functions are programmable. The functions listed in Table 5-7 on page 5-31 are the default functions pre-programmed by the CT11MPCore control PLD design.

5.3 AXI bus timing specification

This section describes the timing for the CT11MPCore multiplexed AXI system bus.

5.3.1 Core Tile timing and the AMBA 3 AXI Protocol

The worst case timing figures at the CT11MPCore headers are shown in Table 5-8 on page 5-34 and Table 5-9 on page 5-35. You must use these figures as a guideline when designing your own boards.

The system bus on RealView Logic Tiles and baseboards is routed between FPGAs. The exact performance of a system depends on the timing parameters of the baseboard and all tiles in the system. Some allowance also has to be made for clock skew, routing delays and number of modules (that is, loading).

Not all Logic Tile or baseboard FPGA implementations meet the ideal timing parameters, due to the complexity of the design or routing congestion within the device. For this reason, the PLL clock generators on baseboards default to a safe low value that all modules can achieve. See the documentation supplied with your baseboard for details on changing the clock frequency.

A detailed timing analysis involves calculating the input/output delays between modules for all timing parameters. In general, the simplest approach to determine the maximum operating frequency is to increase the frequency of the clock generators until the system becomes unstable.

AXI Clock

The AMBA 3 AXI protocol includes a single clock signal, **ACLK**. All input signals are sampled on the rising edge of **ACLK**. All output signal changes must occur after the rising edge of **ACLK**.

There must be no combinatorial paths between input and output signals on both master and slave interfaces.

——— **Note** ———

The CT11MPCore has three clocks:

CLKOUTDIV This clock drives the on-board AXI port 1 signal multiplexers and is buffered out to the Core Tile headers as **CLK_GLOBAL**.

CLKOUTDIVD This clock, identical to **CLKOUTDIV**, drives the on-board AXI port 0 signal multiplexers.

CLK_GLOBAL This clock is the global AXI clock (**ACLK**).

AXI Reset

The AMBA 3 AXI protocol includes a single active LOW reset signal, **ARESETn**. The reset signal can be asserted asynchronously, but deassertion must be synchronous after the rising edge of **ACLK**.

Note

nSYSPORESET is the CT11MPCore global AXI reset (**ARESETn**).

5.3.2 Timing parameters

The following timings are based on the grouped delays for the CT11MPCore. They include the test chip input and output delay, and the delay due to the AXI bus multiplexing.

Table 5-8 shows the Global timing parameters.

Table 5-8 Global timing parameters

Parameter	Description	Max
T_{clk}	**CLK_GLOBAL** clock frequency	20MHz
T_{isrst}	**nSYSPORESET** deasserted setup time before **CLK_GLOBAL**	5ns

Note

During reset the following interface requirements apply:

- a master interface must drive **ARVALID**, **AWVALID**, and **WVALID** LOW
- a slave interface must drive **RVALID** and **BVALID** LOW.

All other signals can be driven to any value.

A master interface must begin driving **ARVALID**, **AWVALID**, or **WVALID** HIGH only at a rising **CLK_GLOBAL** edge after **nSYSPORESET** is HIGH.

The CT11MPCore uses a multiplexed AXI bus. Figure 5-6 on page 5-35 shows the CT11MPCore multiplexed AXI bus input setup timing.

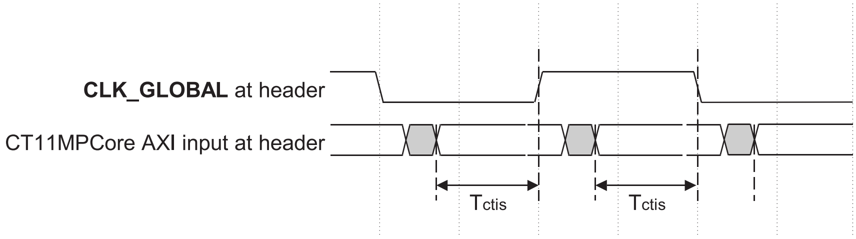


Figure 5-6 CT11MPCore multiplexed AXI bus input setup timing

T_{ctis} Input setup to clock

This is the longest time that the CT11MPCore requires a valid AXI bus signal level to be presented at the Core Tile header before a rising or falling edge of **CLK_GLOBAL**.

Figure 5-7 shows the CT11MPCore multiplexed AXI bus output valid timing.

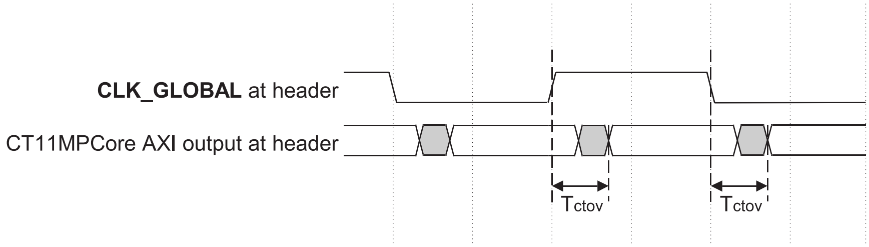


Figure 5-7 CT11MPCore multiplexed AXI bus output valid timing

T_{ctov} Core Tile output valid after clock

This is the longest delay between a rising or falling edge of **CLK_GLOBAL** and a valid AXI bus output signal level arriving at the CT11MPCore header.

Table 5-9 shows the CT11MPCore AXI bus worst case timings when characterized for a slow process, a **VDDCORE** voltage of 1.1V, and a core temperature of 85°C.

Table 5-9 CT11MPCore multiplexed AXI bus timing parameters

Parameter	Description	Max
T_{ctis}	Input setup time to either edge of **CLK_GLOBAL**	8.6ns
T_{ctov}	Output valid time from either edge of **CLK_GLOBAL**	9.4ns

Appendix A
Specifications

This appendix contains the specifications for the CT11MPCore. It contains the following sections:

- *Electrical specification* on page A-2
- *Mechanical details* on page A-4.

Note

See *AXI bus timing specification* on page 5-33 for the timing specifications for Core Tiles that use the HBI-0146 board.

A.1 Electrical specification

This section provides details of the voltage and current characteristics for the CT11MPCore.

A.1.1 Bus interface characteristics

Table A-1 shows the Core Tile electrical characteristics for the system bus interface. The CT11MPCore uses 3.3V and 5V sources.

Table A-1 Core Tile electrical characteristics

Symbol	Description	Min	Max	Unit
3V3	Supply voltage (interface signals)	3.1	3.5	V
5V	Supply voltage (regulators)	4.75	5.25	V
V_{IH}	High-level input voltage	2.0	3.6	V
V_{IL}	Low-level input voltage	0	0.8	V
V_{OH}	High-level output voltage	2.4	-	V
V_{OL}	Low-level output voltage	-	0.4	V
C_{IN}	Input capacitance	-	20	pF

A.1.2 Current requirements

Table A-2 shows current requirements measured at room temperature and nominal voltage.

The power regulators are all powered from the 5V supply. Interface logic on the board is powered from the on board 3.3V.

Table A-2 Current requirements

Component	VDDCORE 1V2	AVDD 1V2	1V8	3V3
MPCore core	500mA	-	-	-
MPCore PLL	-	10mA	-	-

Table A-2 Current requirements (continued)

Component	VDDCORE 1V2	AVDD 1V2	1V8	3V3
MPCore I/O	-	-	-	1A
Control PLD core	-	-	100mA	
Control PLD I/O and Mux logic	-	-	-	500mA

Note

The current requirements shown are typical and are dependant on the application being run.

A.2 Mechanical details

Figure A-1 shows the mechanical outline of the CT11MPCore.

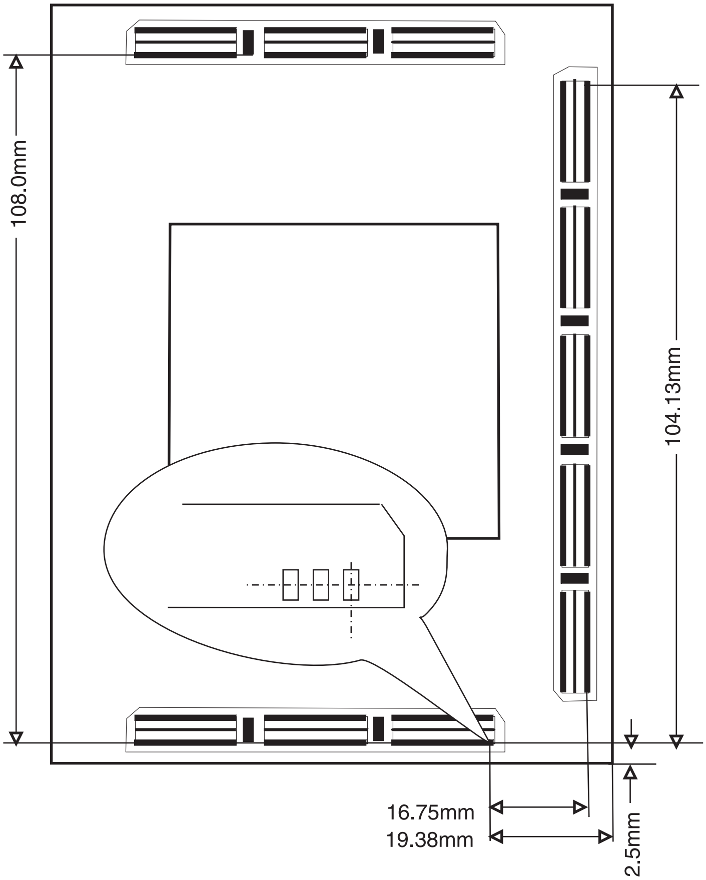


Figure A-1 Board outline (top view)

Glossary

This glossary lists all the abbreviations used in the User Guide.

ADC Analog to Digital Converter. A device that converts an analog signal into digital data.

AXI Advanced eXtensible Interface. An ARM open standard bus protocol.

AMBA 3 Advanced Microcontroller Bus Architecture version 3.

DAC Digital to Analog Converter. A device that converts digital data into analog level signals.

DCC Debug Communications Controller. The DCC is the set of resources that the external DBGTAP debugger uses to communicate with a piece of software running on the core.

DMA Direct Memory Access.

EB RealView Emulation Baseboard. A hardware platform used for system prototyping and debugging of ARM microprocessors.

FPGA Field Programmable Gate Array.

ICE In Circuit Emulator. A interface device for configuring and debugging processor cores.

I/O Input/Output.

JTAG Joint Test Action Group. The committee which defined the IEEE test access port and boundary-scan standard.

LED Light Emitting Diode.

Multi-ICE A system for debugging embedded processor cores using a JTAG interface.

PCI Peripheral Component Interconnect. A circuit board level bus interconnect.

PISMO Memory specification for plug in memory modules.

PLD Programmable Logic Device.

PLL Phase-Locked Loop, a type of programmable oscillator.

RAM Random Access Memory.

RVI RealView ICE. A system for debugging embedded processor cores using a JTAG interface.

SCU Snoop Control Unit. A system block that maintains coherency across the L1 cache memory systems in the ARM11 MPCore.

USB Universal Serial Bus. Hardware interface for connecting peripheral devices.