AHB CPU Wrappers

Technical Reference Manual

ARM®

ARM DDI 0169D

AHB CPU Wrappers
Technical Reference Manual

Release Information

The following changes have been made to this document.

Change history

Date	Issue	Change
18 April 2001	A	First release.
18 May 2001	B	Second release. New Chapter 1 added.
23 June 2003	C	Third release. Changes to Chapter 1.
14 August 2003	D	Corrected product number in Chapter 1.

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`

Contents
AHB CPU Wrappers Technical Reference Manual

Preface

Chapter 1 Introduction

Chapter 2 ARM720T AHB Wrapper

Chapter 3 ARM7TDMI AHB Wrapper

Chapter 4 ARM920T AHB Wrapper

Chapter 5 ARM940T AHB Wrapper

Preface

This preface introduces the *Advanced High-performance Bus* (AHB) *Central Processor Unit* (CPU) Wrappers Technical Reference Manual. It contains the following sections:

- *About this document* on page vi
- *Feedback* on page viii.

About this document

This document is the AHB CPU Wrappers Technical Reference Manual.

Intended audience

This document has been written for experienced hardware and software engineers to help them implement AHB wrappers on various ARM processor cores.

Using this manual

This document is organized into the following chapters:

Chapter 1 ***Introduction***

Read this chapter for a overview of the deliverable directory structure and a summary of the wait states, timing and critical paths, and use of the ETM and coprocessor interfaces.

Chapter 2 ***ARM720T AHB Wrapper***

Read this chapter for a description of the ARM720T and ARM740T core wrappers as used with a AHB-based *Example Amba SYstem* (EASY). It introduces the wrapper block, and describes the AHB signals and the connections to the core.

Chapter 3 ***ARM7TDMI AHB Wrapper***

Read this chapter for a description of the ARM7TDMI and ARM7TDMI-S core wrappers. It introduces the wrapper block, and describes the AHB signals and the connections to the core.

Chapter 4 ***ARM920T AHB Wrapper***

Read this chapter for a description of the ARM920T and ARM922T core wrappers. It introduces the wrapper block, and describes the AHB signals and the connections to the core.

Chapter 5 ***ARM940T AHB Wrapper***

Read this chapter for a description of the ARM940T core wrapper. It introduces the wrapper block, and describes the AHB signals and the connections to the core.

Typographical conventions

The following typographical conventions are used in this book:

italic Highlights important notes, introduces special terminology, denotes internal cross-references, and citations.

bold Highlights interface elements, such as menu names. Denotes ARM processor signal names. Also used for terms in descriptive lists, where appropriate.

`monospace` Denotes text that can be entered at the keyboard, such as commands, file 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 and 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 section lists publications by ARM Limited.

ARM periodically provides updates and corrections to its documentation. See `http://www.arm.com` for current errata sheets, addenda, and the ARM Frequently Asked Questions list.

ARM publications

This document contains information that is specific to the AHB CPU wrappers. Refer to the following document for other relevant information:

- *AHB Example AMBA System Technical Reference Manual* (ARM DDI 0170).

Feedback

ARM Limited welcomes feedback both on the ARM AHB CPU wrappers, and on the documentation.

Feedback on the AHB CPU wrappers

If you have any comments or suggestions about this product, contact your supplier giving:

- the product name
- a concise explanation of your comments.

Feedback on this document

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

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

General suggestions for additions and improvements are also welcome.

Chapter 1
Introduction

This chapter gives a summary of the wait states, timing and critical paths, use of the ETM and coprocessor interfaces, and the release directory structure for the CPU wrappers. It contains the following sections:

1.1 Delivery overview

The ARM CPU AHB wrapper pack contains four primary directories:

`design` Contains the HDL sources for all wrappers.

`validation` Contains the *AMBA Compliance Testbench* (ACT) components for all wrappers.

`ACT` Contains the ACT-e tool required by the testbenches.

`docs` Contains documents pertaining to the CPU AHB wrappers.

The complete directory structure is shown in Figure 1-1.

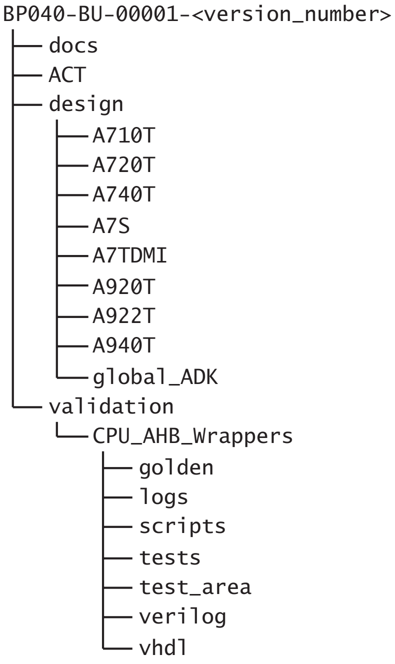

Figure 1-1 Directory structure

Each subdirectory (apart from `global_ADK`) within the `design` directory contains the HDL source files plus simple example synthesis scripts for the relevant CPU. The `global_ADK` directory is part of the ADK product, and contains the core synthesis scripts used by all CPU wrappers.

For a description of the contents of the directories within the validation hierarchy, and for details of using the files within the ACT and validation directories, see the *ARM CPU AHB Wrapper ACT User Guide* which can be found in the `docs` directory.

1.2 CPU bus access wait states

The number of wait states added to CPU bus accesses by each of the CPU AHB wrappers is shown in Table 1-1.

Some wait states added by the AHB wrappers are unavoidable, but others can be masked by wait states from the slave which was accessed in the previous cycle. That is, if the previous access were to a zero-delay slave, the extra wait states added by the AHB wrapper are added to the number of cycles taken to complete the transaction. If the slave previously accessed asserted wait states, no extra cycles are required.

Table 1-1 CPU bus access wait states

CPU AHB wrapper	Added wait state	Reason	Comment
ARM7TDMI and ARM7TDMI-S	1	For each NONSEQ access from the core (except SWP accesses)	These are masked by wait states from the slave being accessed in the previous access cycle.
	2	At the start of a SWP instruction	-
ARM720T	1	At the start of a SWP instruction	These are masked by wait states from the slave being accessed if a waited access immediately precedes the SWP.
ARM920T	1	For each access in a nonbuffered burst of writes (including single accesses, which appear as bursts of undefined length)	-
	1	At the start of a SWP instruction	-
ARM940T	1	At the start of each cacheable/ bufferable burst	-
	1	For each beat of a nonbuffered burst write	These are masked by wait states from the slave being accessed EXCEPT for the final wait state added at the end of the burst.
	1	For each beat of a noncacheable burst read	These are masked by wait states from the slave being accessed EXCEPT for the initial wait state added at the beginning of the burst.
	2	For the read access of a SWP instruction	-
	1	For the write access of a SWP instruction	-

1.2.1 Timing and critical paths

This section describes the timing and critical paths for the various CPU wrappers.

Timing

The desired timing for I/O pins on the AHB is as follows:

- all outputs to be valid within 40% of **HCLK** period after the rising edge of **HCLK**
- all inputs to require no more than 30% of **HCLK** period as setup before the rising edge of **HCLK**.

To minimize the number of wait states added when converting from the native core bus interface to AHB, some combinatorial paths from core pins to the AHB exist. Because of the large output delays and input setup times of some core pins, this can lead to non-ideal timing on the AHB interface. Therefore, although each CPU AHB wrapper can be synthesized to meet the maximum possible frequency of the attached core, doing so degrades timings on the AHB.

The synthesis of the AHB wrappers performed at ARM used the following conditions, each using the Avant! CB25 (0.25um) cell library:

- For ARM7TDMI, ARM7S, and ARM720, the AHB is synthesized to 50MHz, using a generic core timing library capable of achieving 66MHz
- For ARM920 and ARM940, the AHB is synthesized to 100MHz, using a generic core timing library capable of achieving 125MHz.

Critical paths

This section describes the critical paths for each AHB wrapper for the synthesized timing conditions given in *Timing*. If the frequency of **HCLK** relative to the maximum operating frequency of the core is reduced (for example, using an ARM920 capable of achieving 200MHz attached to a 100MHz AHB bus), the critical paths improve because of core pin timings.

Conversely, if the frequency of **HCLK** is chosen to match the maximum operating frequency of the core (for example, using an ARM7TDMI capable of achieving 88MHz attached to an 88MHz AHB bus), the timing of the critical paths degrade, as a percentage of the **HCLK** period.

The critical paths for each AHB wrapper are:

ARM7TDMI

HTRANSM valid after 50% of **HCLK** period because of output delays on **nMREQ** and **SEQ** plus combinatorial logic.

HWDATAM valid after 36% of **HCLK** period because of delays on **DOUT**.

HREADYM requires 30% of **HCLK** period as setup in order to create clock enable for ARM core

ARM7S **HTRANS** valid after 50% of **HCLK** period because of output delays on **TRANS** plus combinatorial logic.

HWDATAM valid after 35% of **HCLK** period because of delays on **DOUT**.

HREADY requires 30% of **HCLK** period as setup to create clock enable for ARM core

ARM720T **HTRANSM** and **HBUSREQM** are generated from **BTRAN** and **AREQ**, respectively. Each of these signals is generated from the falling edge of **HCLK**. **HTRANSM** and **HBUSREQM** do not become valid until very late in the **HCLK** cycle. **HTRANSM** and **HBUSREQM** are constrained in the supplied synthesis scripts, and are valid within 80% of **HCLK** period after the rising edge of **HCLK**.

HWDATAM valid after 50% of **HCLK** period because of output delays on **BD**.

ARM920T **HADDRM** valid after 42% of **HCLK** period because of complex combinatorial logic.

HTRANSM valid after 48% of **HCLK** period because of output delays on **NCMAHB** and **LOK**.

HREADYM requires 30% of **HCLK** period as setup because of complex combinatorial paths.

1.3 Use of ETM and coprocessor interfaces

The CPU AHB wrappers are designed to interface purely between the bus interface of the CPU and the AHB bus (the exception is the ARM7TDMI AHB wrapper which also contains test logic for all core pins). All other interfaces on the CPUs are unaffected by the AHB wrappers. Because of this, interfacing to the ETM or coprocessor ports must be carried out in exactly the same way as for a design which does not include the AHB wrapper.

Each AHB wrapper HDL release includes a top-level file which instantiates both the ARM CPU core and the AHB wrapper. These top-level files are intended as examples only, and are provided to enable use of the supplied synthesis scripts. If used in a design, they must be modified to allow tie-off and connection of the non-AHB wrapper related pins as appropriate.

Chapter 2
ARM720T AHB Wrapper

This chapter describes the ARM720T and ARM740T processor core wrappers that you can use with an AHB-based EASY system. It contains the following sections:

2.1 About the ARM720T AHB wrapper

The ARM720T AHB wrapper module interfaces between the ARM720T core and the AHB. This wrapper can also be used with the following cores:

- ARM710T
- ARM740T.

The top-level block diagram is shown in Figure 2-1, which shows how the wrapper interfaces to the ARM720T. The AHB input signals are routed through the wrapper before becoming inputs to the ARM720T. The outputs are also routed through the wrapper before being driven onto the AHB.

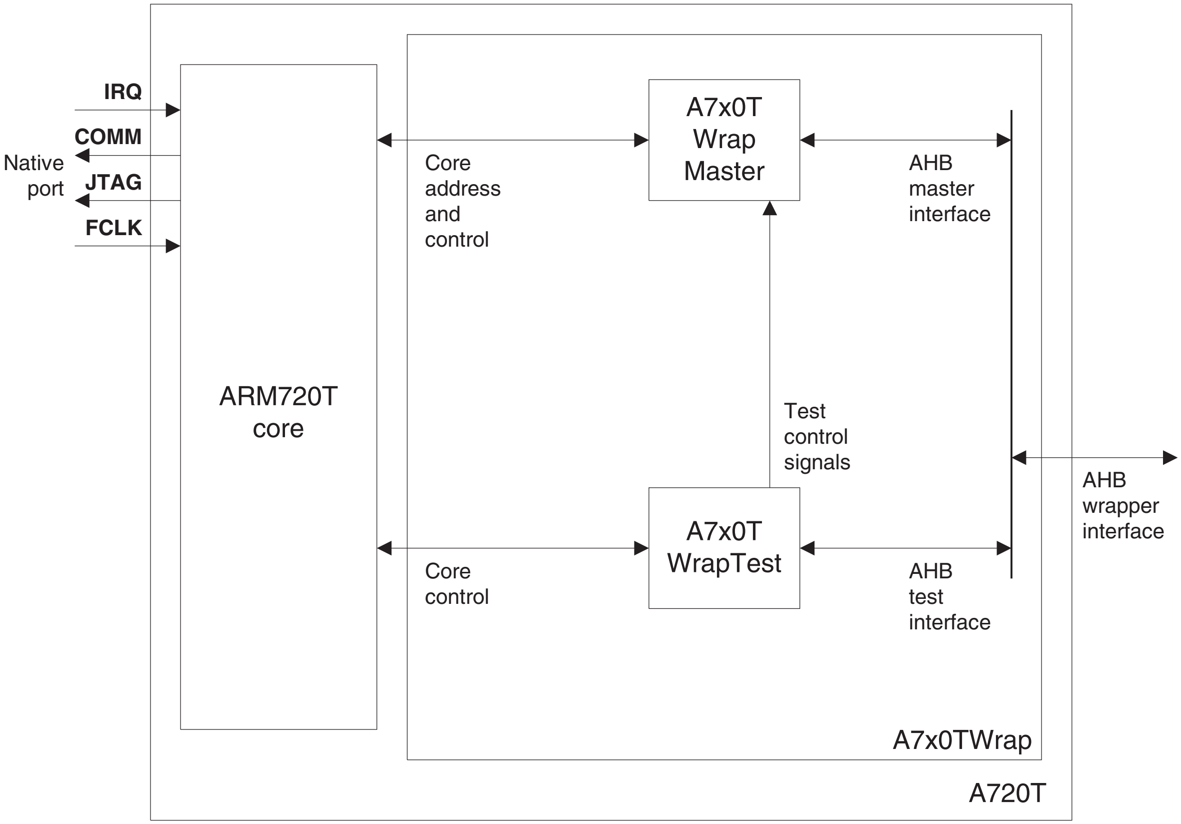

Figure 2-1 ARM720T AHB AMBA wrapper block diagram

2.2 Signal interface

The ARM720T AHB wrapper contains both AHB master and AHB slave interfaces. The master interface is used during normal system operation. The slave interface is used during testing of the ARM720T when the *Test Interface Controller* (TIC) is acting as the current AHB bus master.

This section describes the signal interface of A720TWrap. It does not describe the ARM720T. Only those signals from the ARM720T core that are used by the AHB wrapper are described. All other core signals (for example, coprocessor interface and interrupt inputs) must be connected to the external system in the same way as for an ASB-native design. See the *ARM720T Datasheet* for the relevant information.

Table 2-1 describes the signals used by the ARM720T AHB wrapper.

Table 2-1 ARM720T AHB signal descriptions

Signal	Direction	Description
System inputs		
HCLK	Input	Bus clock. This clock times all bus transfers. All signal timings are related to the rising edge of **HCLK**.
HRESETn	Input	Reset. The bus reset signal is active LOW and is used to reset the system and the bus. This is the only active LOW AHB signal.
Master inputs		
HRDATAM[31:0]	Input	Read data bus. Used to transfer data to the ARM720T in master mode.
HREADYM	Input	Transfer done. When HIGH the **HREADYM** signal indicates that a transfer has finished on the bus. This signal can be driven LOW to extend a transfer.
HRESPM[1:0]	Input	Transfer response. Indicates an OKAY, ERROR, RETRY, or SPLIT response.
HGRANTM	Input	Bus grant. Indicates that the ARM720T is currently the highest priority master. Ownership of the address/control signals changes at the end of a transfer when **HREADYM** is HIGH, so a master gains access to the bus when both **HREADYM** and **HGRANTM** are HIGH.
Master outputs		
HADDRM[31:0]	Output	This is the 32-bit system address bus.
HTRANSM[1:0]	Output	Transfer type. Indicates the type of the current transfer, which can be NONSEQUENTIAL, SEQUENTIAL, or IDLE.
HWRITEM	Output	Transfer direction. When HIGH this signal indicates a write transfer and when LOW a read transfer.

Table 2-1 ARM720T AHB signal descriptions (continued)

Signal	Direction	Description
HSIZEM[2:0]	Output	Transfer size. Indicates the size of the transfer, which can be byte (8-bit), halfword (16-bit), or word (32-bit).
HBURSTM[2:0]	Output	Burst type. Indicates if the transfer forms part of a burst. The ARM720T only performs incrementing bursts of type INCR.
HPROTM[3:0]	Output	Protection control. These signals indicate if the transfer is an opcode fetch or data access, and if the transfer is a Supervisor mode access or User mode access. This is not supported by the ARM720T wrapper, and is always `0000`.
HWDATAM[31:0]	Output	Write data bus. Used to transfer data from the ARM720T in master mode.
HBUSREQM	Output	Bus request. A signal from the wrapper to the bus arbiter which indicates that it requires the bus.
HLOCKM	Output	Locked transfers. When HIGH this signal indicates that the master requires locked access to the bus and no other master must be granted the bus until this signal is LOW.
Slave inputs		
HSELS	Input	This signal selects the ARM720T as slave.
HWRITES	Input	Transfer direction. When HIGH this signal indicates a write transfer and when LOW a read transfer.
HTRANS1S	Input	Transfer type. This is attached to bit 1 of the AHB **HTRANS[1:0]** bus. It indicates an active (NONSEQ or SEQ) or inactive (IDLE or BUSY) transfer.
HWDATAS[31:0]	Input	Write data bus. Used to transfer data to the ARM720T in slave mode.
HREADYS	Input	Transfer done. Indicates that the current transfer on the AHB is finished.
Slave outputs		
HRDATAS[31:0]	Output	Read data bus. Used to transfer data from the ARM720T in slave mode.
HREADYOUTS	Output	Transfer done. In test mode, this signal indicates when a transfer can complete.
HRESPS[1:0]	Output	Transfer response. Indicates a fixed OKAY response.

Table 2-2 on page 2-5 describes the signals to the ARM720T core.

Table 2-2 Signals to ARM720T core

Signal	Direction	Description
BCLK	Output	Bus clock. This times all transfers to and from the ARM720T.
BnRES	Output	Bus reset signal which is active LOW.
BERROR	Output	Error response. Not supported. **BERROR** is LOW and a fixed OKAY response is supplied.
BLAST	Output	Last response. Not supported. **BLAST** is LOW and a fixed continue burst response is supplied.
BWAIT	Output	Wait response. Fixed for zero wait state response, **BWAIT** is LOW. Wait states are controlled by gating **BCLK** to the ARM720T.
AREQ	Input	Bus request. Indicates the ARM720T is requesting the bus.
BA[31:0]	Input	This is the 32-bit system address bus.
BLOK	Input	Locked transfer. The ARM720T requires a locked bus transfer during a SWP instruction.
BSIZE[1:0]	Input	Transfer size. Indicates the size of the transfer, which is either byte (8-bit), halfword (16-bit) or word (32-bit).
BTRAN[1:0]	Input	Transfer type. Indicates the type of the current transfer, which can be NONSEQUENTIAL, SEQUENTIAL, or ADDRESS-ONLY.
BWRITE	Input/output	Transfer direction. When HIGH this signal indicates a write transfer and when LOW a read transfer. During test this signal is driven by the wrapper. During normal operation, it is driven by the ARM720T core.
BD[31:0]	Input/output	Data bus. For master mode write **BD** is connected to **HWDATAM** and read **BD** is connected to **HRDATAM**. For test mode write **BD** is connected to **HWDATAS** and read **BD** is connected to **HRDATAS**.
AGNTarm	Output	Bus grant. In test mode this signal degrants the ARM720T the use of the bus. This is set LOW by the test wrapper state machine.
DSEL	Output	Slave select. In test mode this signal selects the ARM720T as slave. This is set HIGH by the test wrapper state machine.

2.3 Description of the ARM720T wrapper blocks

This section contains descriptions of the following blocks:

- *A720T*
- *A7x0TWrap*
- *A7x0TWrapMaster*
- *A7x0TWrapTest* on page 2-11.

2.3.1 A720T

This is the top-level wrapper which contains A7x0TWrap, and the ARM720T core instantiated as a C model. This wrapper is connected to all native core signals that are not part of the AMBA bus. This allows the wrapper to be easily extracted after RTL simulation for logic synthesis.

2.3.2 A7*x*0TWrap

This wrapper is used to bind the A7x0TWrapMaster block and the A7x0TWrapTest block. It provides a two-port interface that connects, on one side both the master and test mode interfaces directly to the AHB system, and on the other side it connects directly to the native ASB interface of the core.

2.3.3 A7*x*0TWrapMaster

This block handles all the bus interface requirements when the ARM720T is acting as a bus master.

There are eight main sections to this block:

- *Main state machine*
- *Address generation* on page 2-9
- *HTRANSM generation* on page 2-10
- *HLOCKM generation* on page 2-10
- *Clock generation* on page 2-10
- *Holding registers* on page 2-10
- *BD generation* on page 2-11
- *BWAIT, BERROR, and BLAST generation* on page 2-11.

Main state machine

The main state machine controls the majority of the functionality of the wrapper and indicates:

- whether the master is granted the use of the bus

- whether the ARM720T needs to perform a transfer
- when a transfer has received a SPLIT or RETRY response.

Figure 2-2 shows the main state machine.

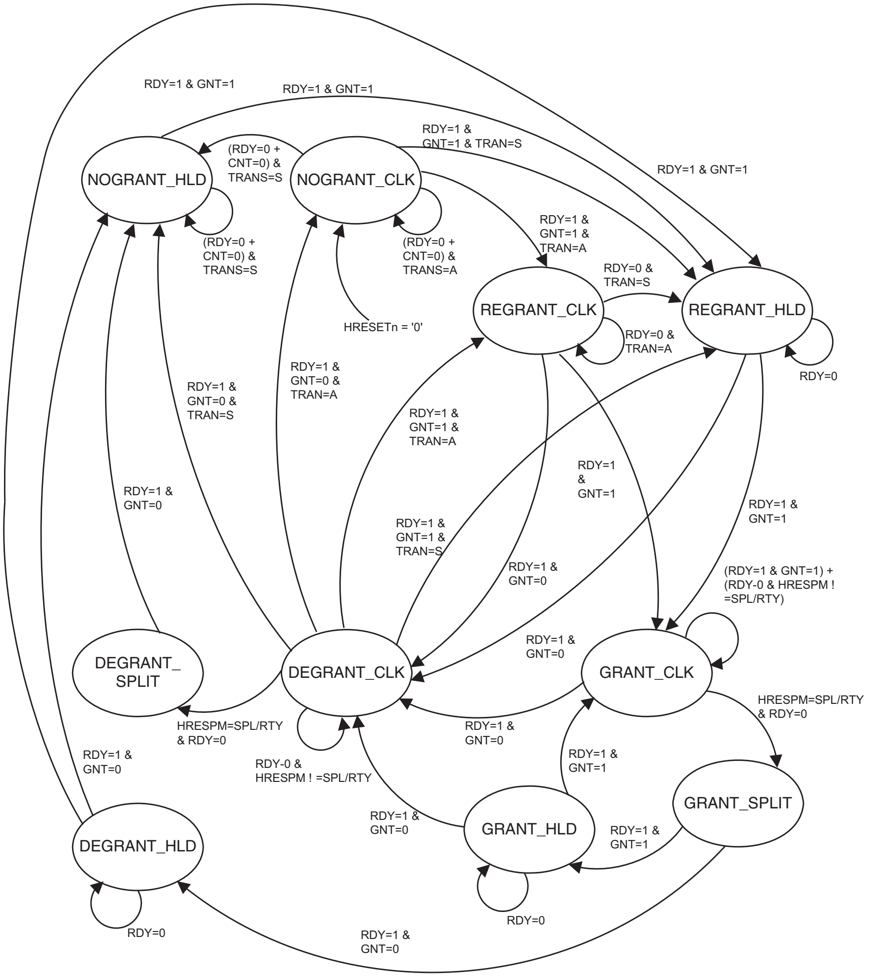

Figure 2-2 ARM720T AHB wrapper main state machine

There are ten states:

NOGRANT_CLK

This state indicates that the ARM720T is not granted use of the bus, but as it does not require the bus to perform a transfer, it can be clocked. When in this state, if the ARM720T has to perform a transfer, it moves to the NOGRANT_HLD state to prevent clocking until it is granted use of the bus.

NOGRANT_HLD

In this state the ARM720T is not granted use of the bus and wishes to perform a transfer, so it is not clocked. The ARM720T remains in this state until it is granted the bus and moves to the REGRANT_HLD state.

REGRANT_CLK

The two REGRANT states are entered when the ARM720T is first granted use of the bus and the ARM720T has ownership of the address/control signals, but does not yet have ownership of the data bus. In the REGRANT_CLK state the ARM720T does not have to perform a transfer and therefore it can be clocked.

REGRANT_HLD

The REGRANT_HLD state is similar to the REGRANT_CLK state as it is entered when the ARM720T is first granted the bus. However, in this state the ARM720T has to perform a transfer and therefore cannot be clocked.

GRANT_CLK

The GRANT_CLK state is the usual state when the ARM720T is granted the bus. The ARM720T is clocked whenever the **HREADYM** signal is HIGH and is not clocked when wait states are inserted on the bus by the slave driving **HREADYM** LOW.

The GRANT_CLK state is exited in two ways:

- if the ARM720T loses ownership of the bus then the state machine moves to the DEGRANT_CLK state
- if a transfer that is being performed receives either a SPLIT or RETRY response then the GRANT_SPLIT state is entered.

GRANT_SPLIT

This state is entered when a transfer receives either a SPLIT or RETRY response. In this state the wrapper forces an IDLE transfer on to the bus to cancel the following transfer. The GRANT_SPLIT state is only ever occupied for a single cycle and is followed either by the GRANT_HLD state if the ARM720T remains granted on the bus. Alternatively, it exits to the DEGRANT_HLD state if ownership of the bus is lost.

GRANT_HLD

This state indicates second stage of a SPLIT or RETRY transfer. In this state the wrapper drives out the address and control signals in the holding register in order to restart the transfer that received the SPLIT or RETRY response.

DEGRANT_CLK

The DEGRANT states indicate the ARM720T is losing ownership of the bus, but the last access is still completing. In the DEGRANT_CLK state the ARM720T is clocked according to the level of the **HREADYM** signal.

DEGRANT_SPLIT

This state is only entered when the last transfer on the bus receives a SPLIT response.

DEGRANT_HLD

This state is only entered when ownership of the bus is lost just after a transfer receives either a SPLIT or RETRY response. In this state the ARM720T is not clocked as it must repeat the transfer that received a SPLIT or RETRY response before it can be clocked.

Address generation

The address generation logic has three sources for the address of the next transfer:

- A registered version of the address from the ARM720T, **BA**. This source of the address is used when the ARM720T performs an ADDRESS-ONLY transfer and effectively reloads the address generation logic with a new address.
- The same as the current transfer. This address source is used when wait states are inserted on the bus.
- An incremented version of the current address. The size of the increment (either +2 or +4) depends on the size of the current transfer. This source of the address is used when the ARM720T is performing a SEQUENTIAL transfer.

HTRANSM generation

The **HTRANSM** signals are generated from the **BTRANS** signals from the ARM720T. The signals from the ARM720T are modified when:

- the transfer source is from the holding registers then the transfer type is forced to be NONSEQUENTIAL
- the current transfer receives either a SPLIT or RETRY response then the next transfer is canceled by forcing **HTRANSM** to IDLE
- the current transfer is the first in a burst then the transfer type is forced to NONSEQUENTIAL.

HLOCKM generation

The **HLOCKM** signal on the bus is generated directly from the **BLOK** signal. The ARM is stalled during the first cycle of a locked transfer to ensure that the **HLOCKM** signal is always asserted a cycle prior to the locked transfer being started on the bus. This is required to meet the AHB specification and allow sufficient time for the arbiter to ensure that the ARM720T has locked access to the bus.

Clock generation

The clock generation logic is used to modify the **BCLK** signal to the ARM720T. The ARM720T clock is removed when:

- the current transfer is waited because the slave has driven **HREADYM** LOW
- the current transfer receives either a SPLIT or RETRY response
- the ARM is not granted the bus, but wishes to perform a transfer.

Holding registers

The address and control signal holding registers are used to store the address and control information for the current transfer. This information is required if the transfer receives either a SPLIT or RETRY response and must be restarted.

The holding registers are also used when the ARM720T loses ownership of the bus. In this case the core has to be clocked to complete the final transfer on the bus, but the address from the ARM720T is not driven onto the bus for the next transfer. Therefore the holding registers are used to store the address until the ARM720T is next granted use of the bus.

BD generation

Because the ARM720T has a single bidirectional data bus, the wrapper is required to combine the AHB read and write data buses. The wrapper does this by only driving the read data, **HRDATAM**, on to **BD** during read transfers. At all other times during normal operation the wrapper does not drive **BD**, avoiding bus clash when the ARM720T drives data on to **BD** for write transfers.

BWAIT, BERROR, and BLAST generation

During normal operation **BWAIT**, **BERROR**, and **BLAST** are all driven LOW. These signals are not required because:

- the wait state control is provided by gating the main clock, **BCLK**, to the ARM720T
- the AHB wrapper does not support ERROR responses from the system, because the ARM720T is expected to make use of the memory management and protection unit provided with the ARM720T.

These signals are all driven by tristate buffers so that they can be set to high impedance during test mode when the ARM720T becomes a bus slave and drives these signals.

2.3.4 A7*x*0TWrapTest

This block handles all the bus interface requirements when the ARM720T is operating in test mode. Its main functions are to ensure a clean entry and exit from the test mode, and to maintain synchronization between the TIC interface and the embedded test mode state machine within the ARM720T.

There are two main sections to this block:

- *Test state machine*
- *Test logic output* on page 2-14.

Test state machine

A test state machine is used as the central controller to provide control of the AHB bus for all test mode transaction types.

Figure 2-3 on page 2-12 shows the test state machine.

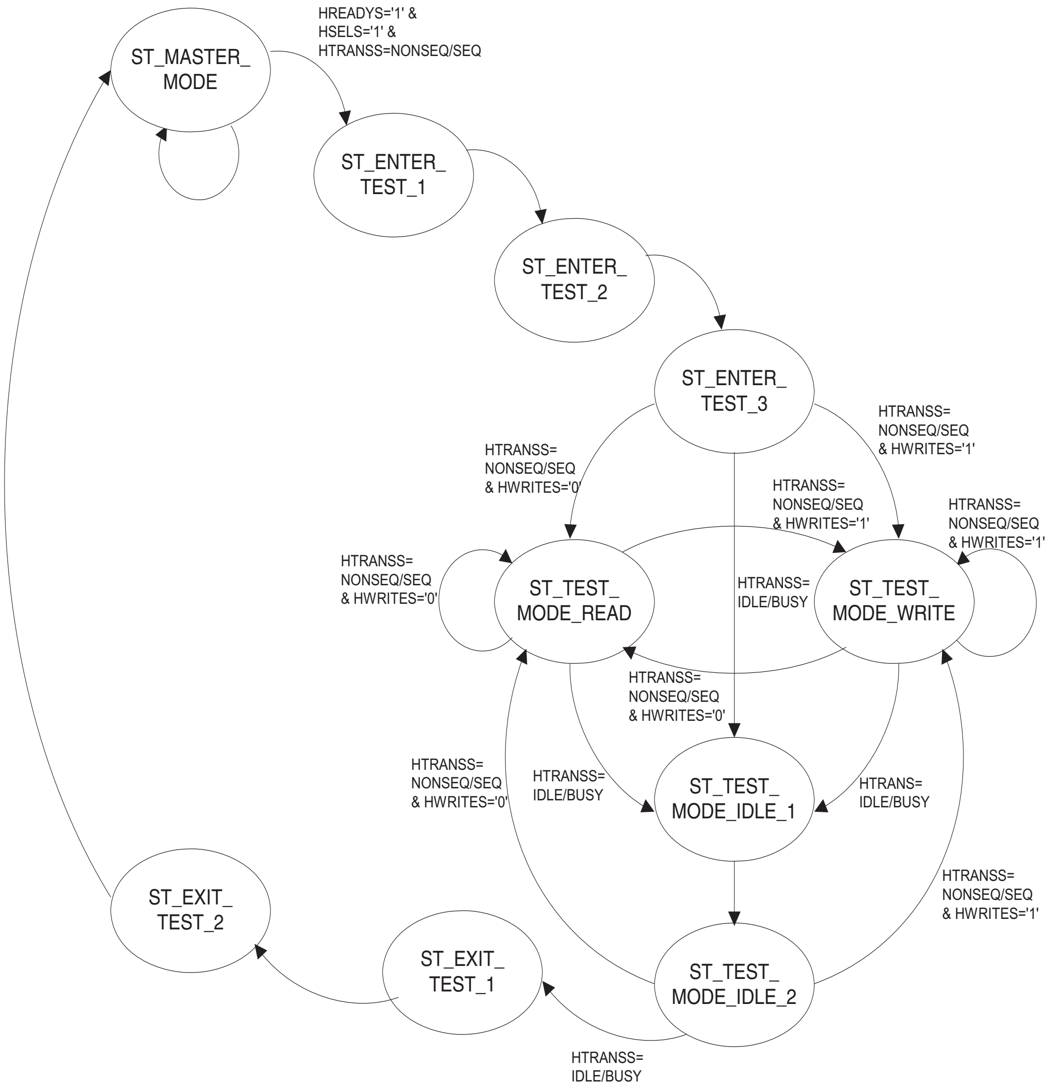

Figure 2-3 Test state machine

There are ten states in the test state machine:

ST_MASTER_MODE

This is the IDLE state for the test state machine. During this state the ARM720T is acting as a bus master and the test logic is disabled. This state is exited when the first transfer of a test sequence is detected.

ST_ENTER_TEST_1

This is the first state on the entry into test and during this state the **AGNTarm** signal is removed from the ARM720T.

ST_ENTER_TEST_2

During the second state on entry in to test the ARM720T clock is enabled so that the ARM720T becomes degranted and tristates all the output signals that must become inputs for test.

ST_ENTER_TEST_3

During the third state on entry in to test the **BWRITE** signal to the ARM720T becomes driven from the test wrapper.

ST_TEST_MODE_WRITE

This state is entered whenever a WRITE transfer to the ARM720T is being performed. During this state the **HWDATAS** is driven on to the bidirectional data bus (**BD**) of the ARM720T.

ST_TEST_MODE_READ

In this state a READ transfer from the ARM720T occurs. During this state the bidirectional data bus (**BD**) of the ARM720T is driven on to **HRDATAS**.

ST_TEST_MODE_IDLE_1

This state is the first of two IDLE states that occur when the ARM720T is moving between READ and WRITE transfers.

ST_TEST_MODE_IDLE_2

This state is the second of two IDLE states that occur when the ARM720T is moving between READ and WRITE transfers. If this state is followed by another IDLE transfer then this indicates the end of the test process and the next state will be ST_EXIT_TEST_1, otherwise this state is followed by a READ or WRITE transfer.

ST_EXIT_TEST_1

In the first exit from test state, the signals that are driven from the wrapper during test become tristate.

ST_EXIT_TEST_2

In the second exit from test state, the ARM720T is regranted as a bus master by driving **AGNTarm** HIGH. After this state the ARM720T returns to the ST_MASTER_MODE state which indicates that the ARM720T has returned to normal operation.

Test logic output

The output signals generated from the test state machine are:

AGNTarm This signal is driven LOW throughout the test process to degrant the ARM720T so that it can be tested as a slave.

TestEnable This signal is used to clock the core throughout the test process. The ARM720T is always clocked except in the ST_TEST_MODE_IDLE_2 state.

SelArmTest The tristate control of signals, such as **BWAIT**, **BERROR**, and **BLAST** is controlled by the **SelArmTest** signal. This ensures that the wrapper does not drive these signals during test when the ARM720T is driving them to respond to slave transfers.

DSEL The **DSEL** select signal to the core is generated from the test state machine before the ST_TEST_MODE_READ or ST_TEST_MODE_WRITE states. A transparent latch is used in the generation of the **DSEL** signal to allow for designs that have late **HTRANSS** timing. In low speed designs it is possible to replace this latch with a falling edge register.

BWRITE The write signal to the ARM720T is generated from the **HWRITES** signal and is a tristate signal which is only driven during test. The write signal must be timed with reference to the rising edge of **BCLK** and therefore a falling edge **HCLK** register must be used to generate this signal.

BD The bidirectional data bus to the ARM720T is driven with the write data, **HWDATAS**, during write transfers to the ARM720T.

HREADYOUTS

The **HREADYOUTS** signal is used to insert wait states on the bus during transfers to the test wrapper. Wait states are only required during the entry in to test and at all other times during the test process transfers receive a zero wait state OKAY response.

HRESPS The response signals are fixed to provide an OKAY response.

HRDATAS During test read transfers the data from **BD** must be driven on to **HRDATAS**. At all other times this bus is driven LOW to prevent unnecessary toggling.

2.4 Non-standard design practices

The following non-standard design practices are described:

- *Clock gating*
- *Transparent latches*
- *Tristate drivers* on page 2-16.

2.4.1 Clock gating

A clock inverter is instantiated within A7x0TWrap (instance name *uClockInv*) to provide **nHCLK** (inverted **HCLK**). This signal is used in the following places:

A7x0TWrapMaster
Creation of **BclkEn**, the enable term for **BCLK**.

A7x0TWrapTest Creation of DriveBwrite and Write, the tristate enable and value used to create **BWRITE**.

An inverted version of the clock is used so that all HDL code describes only rising-edge sequential logic. This is done because some cell libraries do not contain falling-edge registers, and also to avoid possible insertion of unwanted clock gating during synthesis.

A clock NAND gate is used within A7x0TWrapMaster (instance name *uClockNand*) to create **BCLK**, the bus clock for the ASB interface of the ARM720T.

The clock gates are described in the design block ClockInv and ClockNand. The methodology chosen within the supplied synthesis scripts synthesizes these blocks first, sets them as *dont_touch*, and then links them in when synthesizing the wrapper. This ensures that the clock gate instances have known references, and also prevents the optimization routines during later synthesis from altering the gates.

2.4.2 Transparent latches

A transparent latch is instantiated within A7x0TWrapMaster (instance name *uLATS*). This is used to create **BclkEn**, the enable term for **BCLK**. The latch operation is described in the design block LATS (transparent latch with asynchronous set).

A transparent latch is instantiated within A7x0TWrapTest (instance name *uLATR*). This is used to create **DSEL**, the slave select signal for the ARM720T core. The latch operation is described in the design block LATR (transparent latch with asynchronous reset).

The synthesis methodology for the latch blocks is the same as for the clock gates described above.

2.4.3 Tristate drivers

Tristate drivers are used within the wrapper:

- in A7x0TWrap, a tristate driver can be found for **BDOUT**
- in A7x0TWrapTest, tristate drivers can be found for **BWAIT**, **BLAST**, **BERROR**, and **BWRITE**.

Most ASIC design flows require Buskeepers to be added to any signals that might be tristated. For the A7x0T wrapper, the following signals must have Buskeepers added:

- **BA**
- **BLOK**
- **BSIZE**
- **BPROT**
- **BTRAN**
- **BWRITE**
- **BD**
- **BWAIT**
- **BERROR**
- **BLAST**
- **CPdata**.

2.5 Programmer’s model exceptions

The ARM720T AHB wrapper does not support the ERROR response on **HRESP**. This means that instruction and data aborts are not supported.

Chapter 3
ARM7TDMI AHB Wrapper

This chapter describes the ARM7TDMI and ARM7TDMI-S processor core wrapper that you can use with an AHB-based EASY system. It contains the following sections:

3.1 About the ARM7TDMI AHB wrapper

The ARM7TDMI AHB wrapper interfaces between the ARM7TDMI or ARM7TDMI-S core and the AHB bus. The modules that translate access from the core to AHB accesses when the core is the current master are common to both cores. The wrapper itself sits alongside the core, intercepting the memory bus. An example higher-level module is also included for each core. This shows how the core and wrapper can be connected, and is used by the synthesis scripts provided to allow the wrapper to be synthesized alongside a timing file for the core.

For the ARM7TDMI only, the wrapper also allows testing of the ARM7TDMI core when the *Test Interface Controller* (TIC) is the current AHB master, allowing the TIF-format production test vectors supplied by ARM to be used with AHB-based designs. Figure 3-1 shows a top-level block diagram of the ARM7TDMI AHB wrapper.

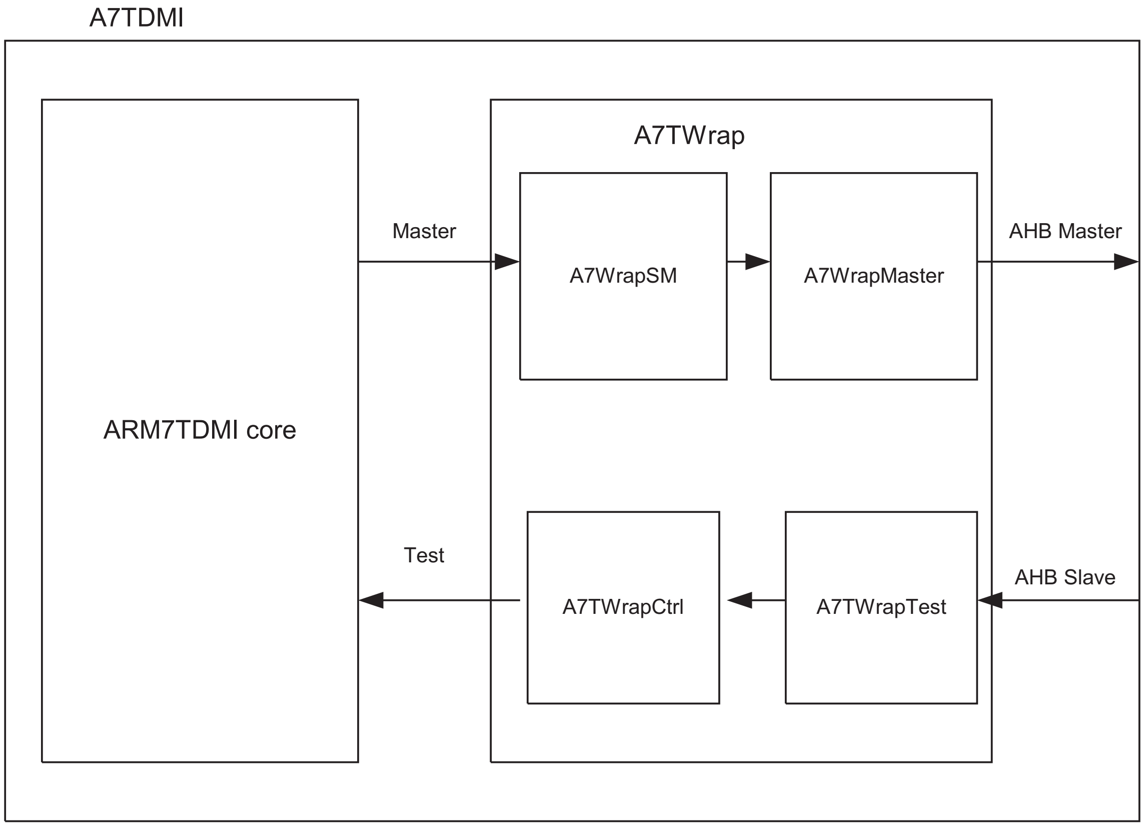

Figure 3-1 ARM7TDMI AHB wrapper block diagram

Figure 3-2 shows a top-level block diagram of the ARM7TDMI-S AHB wrapper.

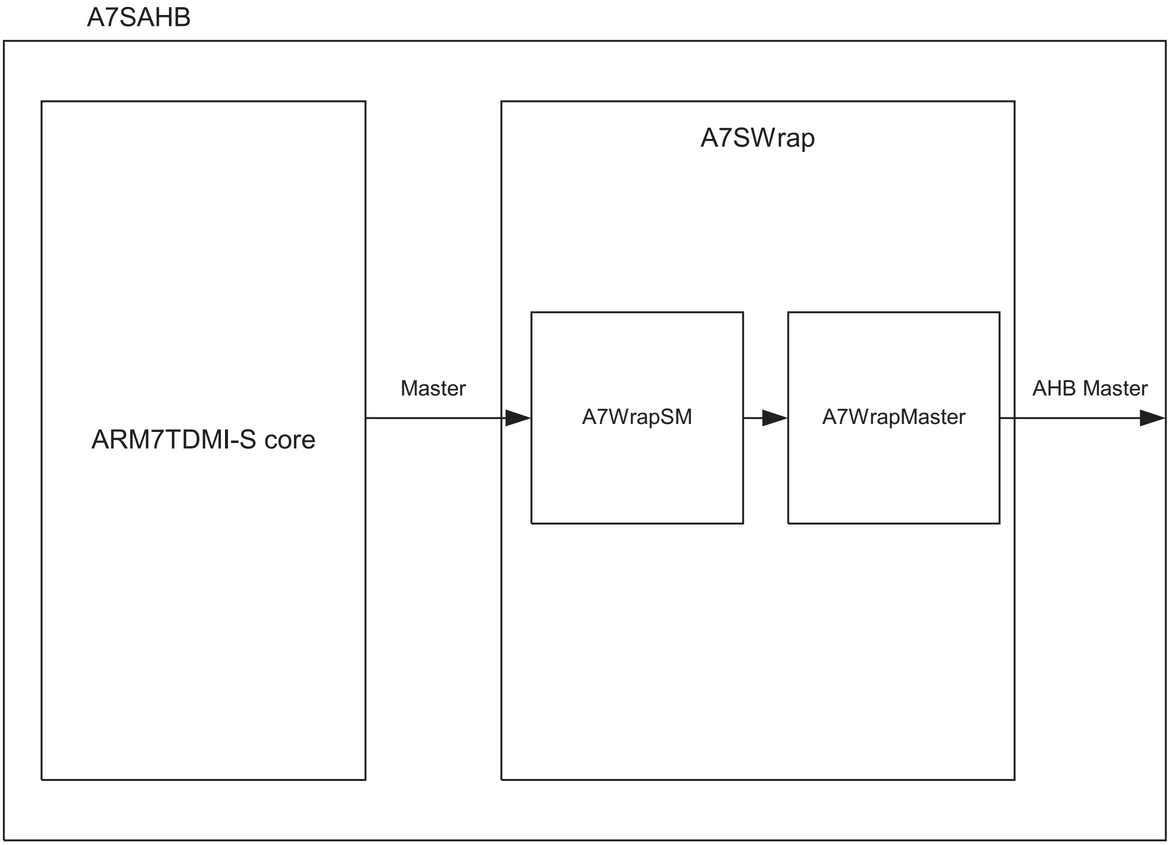

Figure 3-2 ARM7TDMI-S AHB wrapper block diagram

3.2 Signal interface

The ARM7 signals are described in the following sections:

- *A7TWrap signals*
- *A7SWrap signals* on page 3-11.

3.2.1 A7TWrap signals

This section describes the signal interface of A7TWrap. It does not describe the ARM7TDMI. Only those signals from the ARM7TDMI core that are used by the AHB wrapper are described. All other core signals must be connected to the external system in the same way as for a native core design. See the *ARM7TDMI Datasheet* for the relevant information.

Note

The exception is **DOUT[31:0]**, which must be connected directly to the **HWDATA[31:0]** AHB bus and the **DOUT[31:0]** input on A7TWrap.

Table 3-1 describes the signals used by the ARM7TDMI AHB wrapper.

Table 3-1 ARM7TDMI AHB signal descriptions

Signal	Direction	Description
System inputs		
HCLK	Input	Bus clock. This clock times all bus transfers. All signal timings are related to the rising edge of **HCLK**.
HRESETn	Input	Reset. The bus reset signal is active LOW and is used to reset the system and the bus. This is the only active LOW AHB signal.
Master inputs		
HRDATAM[31:0]	Input	Read data bus. Used to transfer data to the ARM7TDMI in master mode.
HREADYM	Input	Transfer done. When HIGH, the **HREADYM** signal indicates that a transfer has finished on the bus. This signal can be driven LOW to extend a transfer.
HRESPM[1:0]	Input	Transfer response. Indicates an OKAY, ERROR, RETRY, or SPLIT response.
HGRANTM	Input	Bus grant. Indicates that the ARM7TDMI is currently the highest priority master. Ownership of the address/control signals changes at the end of a transfer when **HREADYM** is HIGH, so a master gains access to the bus when both **HREADYM** and **HGRANTM** are HIGH.

Table 3-1 ARM7TDMI AHB signal descriptions (continued)

Signal	Direction	Description
Master outputs		
HADDRM[31:0]	Output	This is the 32-bit system address bus.
HTRANSM[1:0]	Output	Transfer type. Indicates the type of the current transfer, that can be NONSEQUENTIAL, SEQUENTIAL, or IDLE.
HWRITEM	Output	Transfer direction. When HIGH, this signal indicates a write transfer and when LOW a read transfer.
HSIZEM[2:0]	Output	Transfer size. Indicates the size of the transfer, that can be byte (8-bit), halfword (16-bit), or word (32-bit).
HBURSTM[2:0]	Output	Burst type. This signal indicates if the transfer forms part of a burst. The ARM7TDMI performs incrementing bursts of type INCR.
HPROTM[3:0]	Output	Protection control. These signals indicate if the transfer is an opcode fetch or data access, and if the transfer is a Supervisor mode access or User mode access.
HBUSREQM	Output	Bus request. A signal from the wrapper to the bus arbiter that indicates that it requires the bus.
HLOCKM	Output	Locked transfer. When HIGH, this signal indicates that the master requires locked access to the bus and no other master must be granted the bus until this signal is LOW.
Slave inputs		
HTRANS1S	Input	Transfer type. This is attached to bit 1 of the AHB **HTRANS[1:0]** bus. It indicates an active (NONSEQ or SEQ) or inactive (IDLE or BUSY) transfer.
HWRITES	Input	Transfer direction. When HIGH, this signal indicates a write transfer and when LOW a read transfer.
HWDATAS[31:0]	Input	Write data bus. Used to transfer data to the ARM7TDMI in slave mode.
HSELS	Input	Slave select. Selects the ARM7TDMI as slave.
HREADYS	Input	Transfer done. When HIGH, this signal indicates that a transfer has finished on the bus. This signal can be driven LOW to extend a transfer.
Slave outputs		
HRDATAS[31:0]	Output	Read data bus. Used to transfer data from the ARM7TDMI in slave mode.
HREADYOUTS	Output	Transfer done. When HIGH, this signal indicates that a transfer to the ARM7TDMI has finished. This signal can be driven LOW to extend a transfer.
HRESPS[1:0]	Output	Transfer response. Indicates an OKAY, ERROR, RETRY, or SPLIT response. The ARM7TDMI always responds with OKAY.

Table 3-2 describes the signals to the ARM7TDMI core.

Table 3-2 Signals to ARM7TDMI core

Signal	Direction	Description
Shared master/slave signals		
nRESET	Output	Active LOW reset to core.
MCLK	Output	Core clock for ARM7TDMI.
DIN[31:0]	Output	Data bus. For master mode this is connected to **HRDATAM**, for test mode it is connected to **HWDATAS**.
DOUT[31:0]	Input	Data bus. For master mode this is connected to **HWDATAM**, for test mode it is connected to **HRDATAS**.
Master inputs		
A[31:0]	Input	Address bus
LOCK	Input	Indicates a locked transfer (such as SWP).
MAS[1:0]	Input	Transfer size. Indicates the size of the transfer, which is typically byte (8-bit), halfword (16-bit) or word (32-bit).
nMREQ	Input	Not memory access.
nOPC	Input	Not opcode access.
nRW	Input	Not read/write.
nTRANS	Input	Not memory translate. When LOW, this signal indicates that the processor is in user mode.
SEQ	Input	Sequential access.
Master outputs		
ABORT	Output	Indicates that an access has aborted (that is, received an ERROR response on the AHB).
nFIQ	Output	Fast interrupt request.
nIRQ	Output	Standard interrupt request.
nWAIT	Output	Indicates that current access is waited.
Test inputs		
BUSDIS	Input	Bus disable. When INTEST is selected on scan chain 0, 4, or 8 this is HIGH. It can be used to disable external logic driving onto the bidirectional data bus during scan testing. This signal changes after the falling edge of **TCK**.

Table 3-2 Signals to ARM7TDMI core (continued)

Signal	Direction	Description
COMMRX	Input	When the communications channel receive buffer is full this is HIGH. This signal changes after the rising edge of **MCLK**.
COMMTX	Input	When the communications channel transmit buffer is empty this is HIGH. This signal changes after the rising edge of **MCLK**.
DBGACK	Input	Debug acknowledge. When the processor is in a debug state this is HIGH.
DBRQI	Input	Internal debug request. This is the logical OR of **DBGRQ** and bit 1 of the debug control register.
HIGHZ	Input	When the **HIGHZ** instruction has been loaded into the TAP controller this signal is HIGH.
RANGEOUT0	Input	EmbeddedICE macrocell. When the EmbeddedICE watchpoint unit 0 has matched the conditions currently present on the address, data, and control buses, then this is HIGH. This signal is independent of the state of the watchpoint enable control bit.
RANGEOUT1	Input	EmbeddedICE macrocell. As **RANGEOUT0** but corresponds to the EmbeddedICE watchpoint unit 1.
nCPI	Input	Not coprocessor instruction. LOW when a coprocessor instruction is processed. The processor then waits for a response from the coprocessor on the **CPA** and **CPB** lines. If **CPA** is HIGH when **MCLK** rises after a request has been initiated by the processor, then the coprocessor handshake is aborted, and the processor enters the undefined instruction trap. If **CPA** is LOW at this time, the processor enters a busy-wait period until **CPB** goes LOW before completing the coprocessor handshake.
nENOUT	Input	Not enable output. During a write cycle, this signal is driven LOW before the rising edge of **MCLK**, and remains LOW for the entire cycle. This can be used to aid arbitration in shared bus applications.
nENOUTI	Input	Not enable output. During a write cycle, this signal is driven LOW before the rising edge of **MCLK**, and remains LOW for the entire cycle. This can be used to aid arbitration in shared bus applications.
nEXEC	Input	Not executed. This is HIGH when the instruction in the execution unit is not being executed because, for example, it has failed its condition code check.
nM[4:0]	Input	Not processor mode. These are the inverse of the internal status bits indicating the current processor mode.
nTDOEN	Input	Not **TDO** enable. When serial data is being driven out on **TDO** this is LOW. Usually used as an output enable for a **TDO** pin in a packaged part.

Table 3-2 Signals to ARM7TDMI core (continued)

Signal	Direction	Description
SCREG[3:0]	Input	Scan chain register. These reflect the ID number of the scan chain currently selected by the TAP controller. These change on the falling edge of **TCK** when the TAP state machine is in the UPDATE-DR state.
TBIT	Input	Thumb state. When the processor is executing the THUMB instruction set, this is HIGH. It is LOW when executing the ARM instruction set. This signal changes in phase two in the first execute cycle of a BX instruction.
xnTRST	Input	JTAG reset input from external tester.
xTCK	Input	JTAG clock input from external tester.
xTDI	Input	JTAG data input from external tester.
xTMS	Input	JTAG mode select input from external tester.
xTDO	Input	JTAG data input from core.
Slave outputs		
ABE	Output	Address bus enable. The address bus drivers are disabled when this is LOW, putting the address bus into a high impedance state. This also controls the **LOCK**, **MAS[1:0]**, **nRW**, **nOPC**, and **nTRANS** signals in the same way. **ABE** must be tied HIGH if there is no system requirement to disable the address drivers.
ALE	Output	Address latch enable. This signal is provided for backwards compatibility with older ARM processors. For new designs, if address retiming is required, ARM Limited recommends the use of **APE**, and for **ALE** to be connected HIGH. The address bus, **LOCK**, **MAS[1:0]**, **nRW**, **nOPC**, and **nTRANS** signals are latched when this is held LOW. This allows these address signals to be held valid for the complete duration of a memory access cycle. For example, when interfacing to ROM, the address must be valid until after the data has been read.
APE	Output	Address pipeline enable. Selects whether the address bus, **LOCK**, **MAS[1:0]**, **nRW**, **nTRANS**, and **nOPC** signals operate in pipelined (**APE** is HIGH) or depipelined mode (**APE** is LOW). Pipelined mode is particularly useful for DRAM systems, where it is desirable to provide the address to the memory as early as possible, to allow longer periods for address decoding and the generation of DRAM control signals. In this mode, the address bus does not remain valid to the end of the memory cycle. Depipelined mode can be useful for SRAM and ROM access. Here the address bus, **LOCK**, **MAS[1:0]**, **nRW**, **nTRANS**, and **nOPC** signals must be kept stable throughout the complete memory cycle. However, this does not provide optimum performance.

Table 3-2 Signals to ARM7TDMI core (continued)

Signal	Direction	Description
BIGEND	Output	Big-endian configuration. Selects how the processor treats bytes in memory: HIGH for big-endian format LOW for little-endian format.
BL[3:0]	Output	Byte latch control. The values on the data bus are latched on the falling edge of **MCLK** when these signals are HIGH. For most designs these signals must be tied HIGH.
BREAKPT	Output	Breakpoint. A conditional request for the processor to enter debug state is made by placing this signal HIGH. If the memory access at that time is an instruction fetch, the processor enters debug state only if the instruction reaches the execution stage of the pipeline. If the memory access is for data, the processor enters debug state after the current instruction completes execution. This allows extension of the internal breakpoints provided by the EmbeddedICE Logic.
BUSEN	Output	Bus enable. A static configuration signal that selects whether the bidirectional data bus (**D[31:0]**) or the unidirectional data buses (**DIN[31:0]** and **DOUT[31:0]**) are used for transfer of data between the processor and memory. When **BUSEN** is LOW, **D[31:0]** is used. **DOUT[31:0]** is driven to a value of zero, and **DIN[31:0]** is ignored, and must be tied LOW. When **BUSEN** is HIGH, **DIN[31:0]** and **DOUT[31:0]** are used. **D[31:0]** is ignored and must be left unconnected.
CPA	Output	Coprocessor absent. Placed LOW by the coprocessor if it is capable of performing the operation requested by the processor.
CPB	Output	Coprocessor busy. Placed LOW by the coprocessor when it is ready to start the operation requested by the processor. It is sampled by the processor when **MCLK** goes HIGH in each cycle in which **nCPI** is LOW.
DBE	Output	Data bus enable. Must be HIGH for data to appear on either the bidirectional or unidirectional data output bus. When LOW the bidirectional data bus is placed into a high impedance state and data output is prevented on the unidirectional data output bus. It can be used for test purposes or in shared bus systems.
DBGEN	Output	Debug enable. A static configuration signal that disables the debug features of the processor when held LOW. This signal must be HIGH to allow the EmbeddedICE Logic to function.

Table 3-2 Signals to ARM7TDMI core (continued)

Signal	Direction	Description
DBGRQ	Output	Debug request. This is a level-sensitive input, that when HIGH causes the ARM7TDMI core to enter debug state after executing the current instruction. This allows external hardware to force the ARM7TDMI core into debug state, in addition to the debugging features provided by the EmbeddedICE Logic.
EXTERN0	Output	External input 0. This is connected to the EmbeddedICE Logic and allows breakpoints and watchpoints to be dependent on an external condition.
EXTERN1	Output	External input 1. This is connected to the EmbeddedICE Logic and allows breakpoints and watchpoints to be dependent on an external condition.
ISYNC	Output	Synchronous interrupts. Set this HIGH if **nIRQ** and **nFIQ** are synchronous to the processor clock, LOW for asynchronous interrupts.
nENIN	Output	Not enable input. This must be LOW for the data bus to be driven during write cycles. Can be used in conjunction with **nENOUT** to control the data bus during write cycles.
nTRST	Output	Not test reset. Reset signal for the boundary-scan logic. This pin must be pulsed or driven LOW to achieve normal device operation, in addition to the normal device reset, **nRESET**.
SDOUTBS	Output	Boundary scan serial output data. Accepts serial data from an external boundary-scan chain output, synchronized to the rising edge of **TCK**. This must be tied LOW, if an external boundary-scan chain is not connected.
TBE	Output	Test bus enable. When LOW, **D[31:0]**, **A[31:0]**, **LOCK**, **MAS[1:0]**, **nRW**, **nTRANS**, and **nOPC** are set to high impedance. Similar in effect as if both **ABE** and **DBE** had been driven LOW. However**, TBE** does not have an associated scan cell and so allows external signals to be driven high impedance during scan testing. Under normal operating conditions **TBE** must be HIGH.
TCK	Output	Test clock. Clock signal for all test circuitry. When in debug state, this is used to generate **DCLK**, **TCK1**, and **TCK2**.
TDI	Output	Test data in. Serial data for the scan chains.
TMS	Output	Test mode select. Mode select for scan chains.

3.2.2 A7SWrap signals

This section describes the signal interface of A7SWrap. It does not describe the A7SAHB. Only those signals from the ARM7TDMI-S core that are used by the AHB wrapper are described. All other core signals must be connected to the external system in the same way as for a native core design. See the *ARM7TDMI-S* datasheet for the relevant information.

The exceptions are:

- **CLK** connects to **HCLK**
- **nRESET** connects to **HRESETn**
- **WDATA[31:0]** connects to **HWDATA[31:0]**
- **RDATA[31:0]** connects to **HRDATA[31:0].**

Table 3-3 describes the signals used by the ARM7TDMI-S AHB wrapper.

Table 3-3 ARM7TDMI-S AHB signal descriptions

Signal	Direction	Description
System inputs		
HCLK	Input	Bus clock. This clock times all bus transfers. All signal timings are related to the rising edge of **HCLK**.
HRESETn	Input	Reset. The bus reset signal is active LOW and is used to reset the system and the bus. This is the only active LOW AHB signal.
AHB inputs		
HREADY	Input	Transfer done. When HIGH the **HREADY** signal indicates that a transfer has finished on the bus. This signal can be driven LOW to extend a transfer.
HRESP[1:0]	Input	Transfer response. Indicates an OKAY, ERROR, RETRY, or SPLIT response.
HGRANT	Input	Bus grant. Indicates that the ARM7TDMI-S is currently the highest priority master. Ownership of the address/control signals changes at the end of a transfer when **HREADY** is HIGH, so a master gains access to the bus when both **HREADY** and **HGRANT** are HIGH.
AHB outputs		
HADDR[31:0]	Output	This is the 32-bit system address bus.
HTRANS[1:0]	Output	Transfer type. Indicates the type of the current transfer, which can be NONSEQUENTIAL, SEQUENTIAL, or IDLE.
HWRITE	Output	Transfer direction. When HIGH this signal indicates a write transfer and when LOW a read transfer.

Table 3-3 ARM7TDMI-S AHB signal descriptions (continued)

Signal	Direction	Description
HSIZE[2:0]	Output	Transfer size. Indicates the size of the transfer, which can be byte (8-bit), halfword (16-bit), or word (32-bit).
HBURST[2:0]	Output	Burst type. Indicates if the transfer forms part of a burst. The ARM7TDMI-S performs incrementing bursts of type INCR.
HPROT[3:0]	Output	Protection control. These signals indicate if the transfer is an opcode fetch or data access, and if the transfer is a Supervisor mode access or User mode access.
HBUSREQ	Output	Bus request. A signal from the wrapper to the bus arbiter which indicates that it requires the bus.
HLOCK	Output	Locked transfers. When HIGH this signal indicates that the master requires locked access to the bus and no other master must be granted the bus until this signal is LOW.

Table 3-4 describes the signals to the ARM7TDMI-S core.

Table 3-4 Signals to ARM7TDMI-S core

Signal	Direction	Description
Inputs from core		
ADDR[31:0]	Input	Address bus.
PROT[1:0]	Input	These output signals to the memory system indicate whether the output is code or data and whether access is User-Mode or privileged access: x0 = opcode fetch x1 = data access 0x = User mode access 1x = Supervisor or privileged mode access.
SIZE[1:0]	Input	Memory access width. These output signals indicate to the external memory system when a word transfer or a halfword or byte length is required: 00 = 8-bit byte access (addressed in word by **ADDR[1:0]**) 01 = 16-bit halfword access (addressed in word by **ADDR[1]**) 10 = 32-bit word access (always word-aligned) 11 = reserved. This signal is analogous to **MAS[1:0]** on the hard macrocell.

Table 3-4 Signals to ARM7TDMI-S core (continued)

Signal	Direction	Description
TRANS[1:0]	Input	Next transaction type. **TRANS** indicates the next transaction type: 00 = address-only (internal operation cycle) 01 = coprocessor 10 = memory access at nonsequential address 11 = memory access at sequential burst address. The **TRANS[1]** signal is analogous to inverted **nMREQ** and the **TRANS[0]** signal is analogous to **SEQ** on the hard macrocell. **TRANS** is analogous to **BTRAN** on the AMBA system bus.
WRITE	Input	Write/read access. When HIGH, **WRITE** indicates a processor write cycle, when LOW, it indicates a processor read cycle. This signal is analogous to **nRW** on the hard macrocell.
LOCK	Input	Locked transaction operation. When **LOCK** is HIGH, the processor is performing a locked memory access. The arbiter must wait until **LOCK** goes LOW before allowing another device to access the memory.
Outputs to core		
ABORT	Output	Memory abort or bus error. This is an input that is used by the memory system to signal to the processor that a requested access is disallowed.
CLKEN	Output	Wait state control. When accessing slow peripherals, the ARM7TDMI-S can be made to wait for an integer number of **CLK** cycles by driving **CLKEN** LOW. When the **CLKEN** control is not used, it must be tied HIGH. This signal is analogous to **nWAIT** on the hard macrocell.

3.3 Description of the ARM7TDMI wrapper blocks

This section contains descriptions of the following blocks:

- *A7TDMI*
- *A7TWrap*
- *A7SAHB*
- *A7SWrap* on page 3-16
- *A7WrapSM* on page 3-16
- *A7WrapMaster* on page 3-19
- *A7TWrapTest* on page 3-19
- *A7TWrapCtrl* on page 3-26.

3.3.1 A7TDMI

This is an example top level containing instantiations of both the ARM7TDMI core and the AHB wrapper. It is used by the provided synthesis scripts to allow synthesis of the wrapper alongside a characterized timing file of the core. All core signals not connected to the AHB wrapper are shown with example connections only. You can change these, or use a completely new top-level in the design.

3.3.2 A7TWrap

This instantiates the master and slave components of the ARM7TDMI AHB wrapper. It also contains some glue logic to make the core appear more like an ARM7TDMI-S. This includes:

- multiplexor on **DIN**
- combinational creation of TRANS from **nMREQ** and **SEQ**
- creation of **MCLK** from inverted **HCLK**
- delayed **CLKEN**.

3.3.3 A7SAHB

This is an example top level containing instantiations of both the ARM7TDMI-S core and the AHB wrapper. It is used by the provided synthesis scripts to allow synthesis of the wrapper alongside a characterized timing file of the core. All core signals not connected to the AHB wrapper are shown with example connections only. You can change these, or use a completely new top-level in the design.

3.3.4 A7SWrap

This instantiates the components of the ARM7TDMI-S AHB wrapper. No slave interface is required for the ARM7TDMI-S, because production testing can be carried out using full scan testing, rather than replay of TIF test vectors through the *Test Interface Controller* (TIC).

3.3.5 A7WrapSM

In master mode, this block converts the memory accesses from the core into pseudo-AHB accesses. This pseudo-AHB format is similar to AHB-Lite. The following sections of the code are explained:

- *Main state machine*
- *Address generation* on page 3-17
- *Wait states* on page 3-18
- *Error support* on page 3-18
- *Control signaling (HTRANS, HSIZE, HBURST, HPROT)* on page 3-18.

Main state machine

The main state machine uses the state variable AddrState (which is registered to create LastAddrState). This can take the following enumerated values:

IDLE Default state. Indicates that no transfer is taking place on the AHB.

IS Indicates that the current IDLE cycle can be part of a merged I-S access.

NON Indicates that the current access is a nonsequential transfer.

SEQ Indicates that the current access is a sequential transfer.

LOKI Pause state to allow **HLOCK** to be asserted on the AHB for one cycle prior to the read access of a SWP instruction.

LOKR Read access of a SWP instruction.

LOKW Write access of a SWP instruction.

Figure 3-3 on page 3-17 shows the possible state transitions. State transitions occur on the rising edge of **HCLK**. Transitions back to the same state can occur either when the access on the pseudo-AHB is waited, or during sequential accesses in a burst (SEQ only).

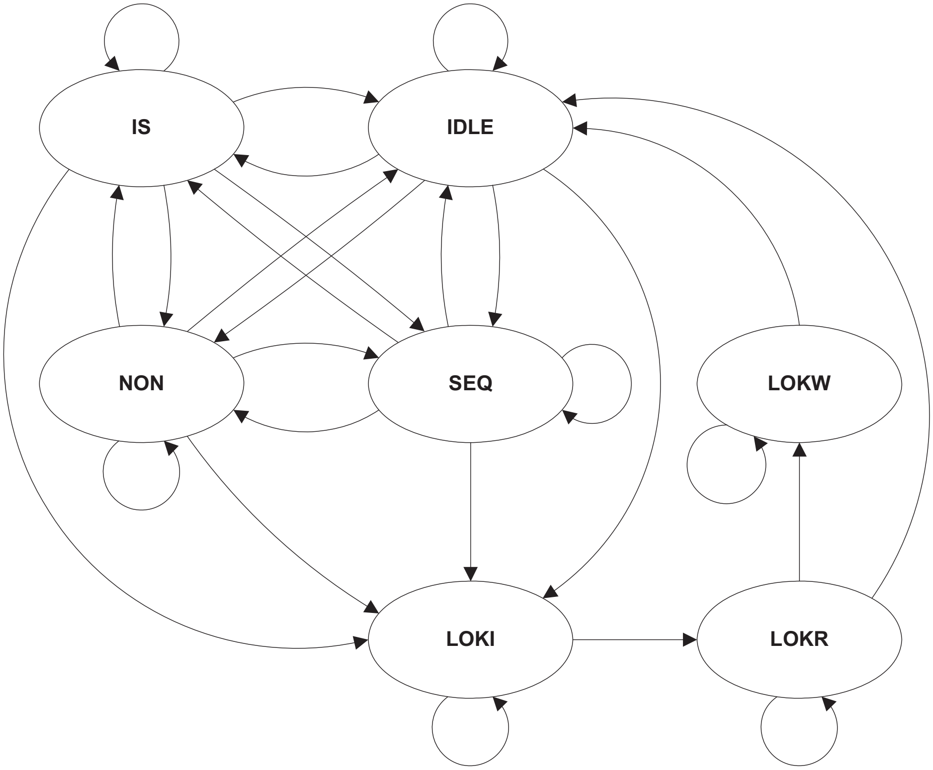

Figure 3-3 ARM7 AHB wrapper main state machine

Address generation

There are two internal sources of address:

- a registered version of the core address
- a locally incremented address created within A7WrapSM.

Normally the registered version of the address is used, but the registered version is required when performing sequential accesses to prevent a combinational path from the core to **HADDR**.

Wait states

Wait states on the AHB (indicated by **HREADY** being LOW) are propagated to the ARM7 only when the core is attempting to access the bus. A string of IDLE cycles will not be waited. Additional wait states (where one wait state equates to a single cycle of **HCLK**) are added as follows:

- +1 wait state for each NONSEQ access from the core (except SWP accesses). These are masked by wait states from the previous access, if any are received. Instruction fetches from the ARM7 cores are always initiated as merged I-S cycles, and so do not see this extra wait state.
- +2 wait states on the read at the start of a SWP instruction. This allows for **HLOCK** to be asserted for a cycle before the AHB access commences.

Error support

The ARM7 AHB wrappers include support for the ERROR response on **HRESP**.

Control signaling (HTRANS, HSIZE, HBURST, HPROT)

Table 3-5 shows the control signal values used by the ARM7 AHB wrapper.

Table 3-5 Control signal values

Signal	Value
HTRANS	00: IDLE 10: NONSEQUENTIAL 11: SEQUENTIAL
HSIZE	00: BYTE 01: HALFWORD 10: WORD
HBURST	001: INCR
HPROT	bit 3: (cacheable) 0 for all accesses bit 2: (bufferable) 0 for all accesses bit 1: 1 = privileged access, 0 = user access bit 0: 1 = data access, 0 = opcode fetch

3.3.6 A7WrapMaster

In master mode, this block converts the pseudo-AHB accesses from A7WrapSM into true AHB by adding support for split and retry responses, and also for **HGRANT**. A SPLIT or RETRY response, or loss of **HGRANT**, causes the affected access to be placed into a holding register for reconstruction when the access can recommence on the AHB. During this time, the pseudo-AHB access is simply waited using **MREADY**.

The holding registers store the current transfer on the pseudo-AHB on each valid cycle (qualified by **MREADY**). The AHB outputs are multiplexed based on the signal **HoldSel**. This is a registered signal, which is synchronously set and cleared by **HoldSet** and **HoldClr** respectively:

- **HoldSet** is activated when a SPLIT or RETRY response is received during the data phase of an access to the core, or when ownership of the AHB bus is lost whilst attempting an active transfer (that is, NONSEQ or SEQ).
- **HoldClr** is activated when **HoldSel** is active, the AHB wrapper has ownership of the bus and any SPLIT or RETRY response has completed.

3.3.7 A7TWrapTest

The test interface block is used to allow the wrapper module to act as an AHB slave during TIC testing of the core. The main parts of this block are:

- the test state machine, that controls the application of the test vectors
- the 28-bit test register, that stores the value of the control inputs during test

The test state machine uses the state variable NextTest (which is registered to create CurrentTest). This can take the following enumerated values:

ST_INACTIVE

Default state. This state is used when the wrapper is not in test mode, and all test outputs are driven to their default levels. The core is clocked as normal in this state.

ST_CTRL_IN

This state is used to load the test register with the control data that is currently on the write data bus. This then determines the values of the control signals that is applied to the core when it is clocked. The core is not clocked during this state.

ST_DATA_IN

In this state write data is being applied to the core (the core is performing a read transfer). The core is clocked in this state.

ST_DATA_OUT

In this state read data is being loaded from the core (the core is performing a write transfer). The core is clocked in this state.

ST_STAT_OUT

This state is used to read the output status signals from the core. The core is not clocked in this state.

ST_ADDR_OUT

This state is used to read the address output from the core. The core is not clocked in this state.

ST_TURNAROUND

This state is used to allow the external data bus time to turnaround between the address read cycle and the control vector write cycle. The core is not clocked in this state.

The state diagram for the test state machine is shown in Figure 3-4 on page 3-21.

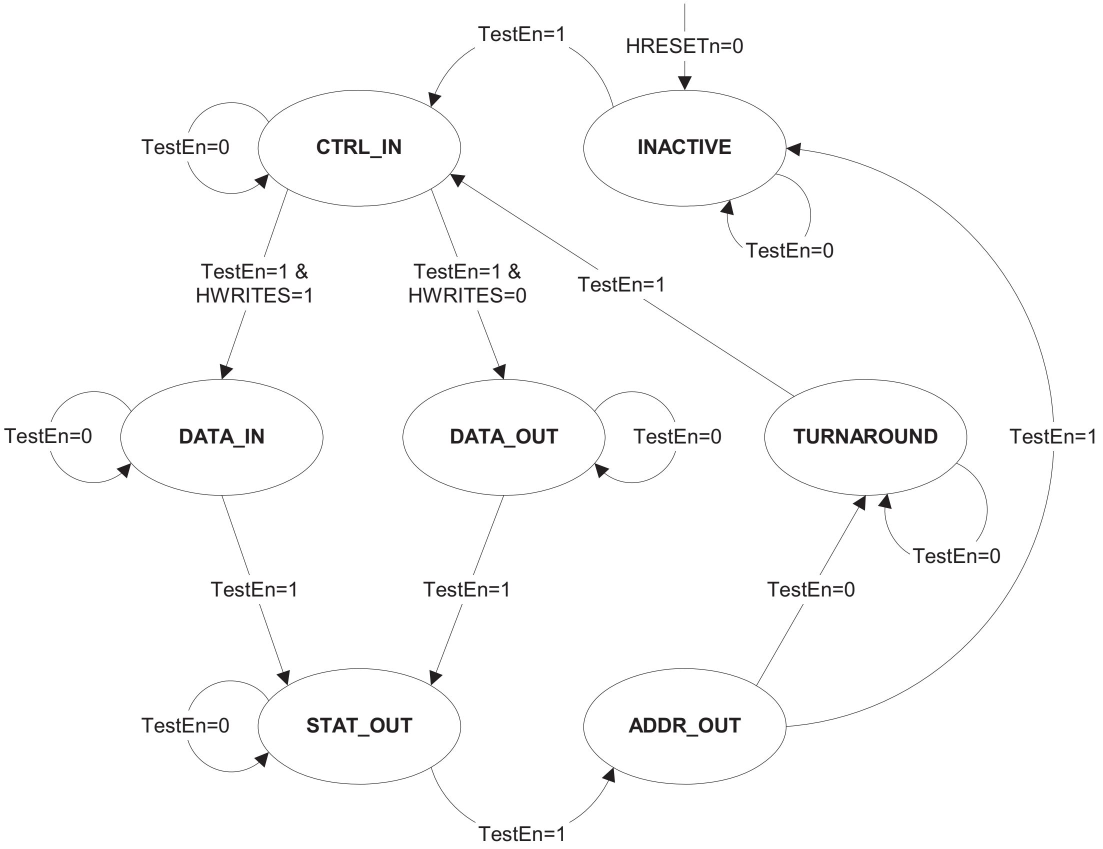

Figure 3-4 ARM7 AHB wrapper test state machine

The **TestEn** signal is used to control when test vectors are applied to the core, and controls the transitions through the test state machine. **TestEn** is set HIGH when the core is addressed during a valid transfer, when the **HTRANS** input indicates a NONSEQUENTIAL or a SEQUENTIAL transfer.

The 28-bit test register that is loaded during the ST_CTRL_IN state determines the control inputs to the core when it is clocked during the ST_DATA_IN or ST_DATA_OUT states. Table 3-6 shows the control input bit positions.

Table 3-6 ARM7TDMI control input bit position

Signal	Description	Bit position	Comments
SDOUTBS	Boundary scan serial output data	27	-
TBE	Test bus enable	26	-
APE	Address pipeline enable	25	-
BL[3:0]	Byte latch control	24:21	ANDed with **TestClk**, and should only be valid during data access cycle.
TMS	Test mode select	20	-
TDI	Test data in	19	-
TCK	Test clock	18	ANDed with **TestClk.**
nTRST	Not test reset	17	-
EXTERN1	External input 1	16	-
EXTERN0	External input 0	15	-
DBGRQ	Debug request	14	-
BREAKPT	Breakpoint	13	-
DBGEN	Debug enable	12	-
ISYNC	Synchronous interrupts	11	-
BIGEND	Big-endian configuration	10	-
CPA	Coprocessor absent	9	-
CPB	Coprocessor busy	8	-
ABE	Address bus enable	7	This must normally be set HIGH, because if the address bus is tristated (**ABE** LOW), then it is not possible to read address values.
ALE	Address latch enable	6	-

Table 3-6 ARM7TDMI control input bit position (continued)

Signal	Description	Bit position	Comments
DBE	Data bus enable	5	
nFIQ	Not fast interrupt request	4	-
nIRQ	Not interrupt request	3	-
ABORT	Memory abort	2	This must normally be driven when **HRESP** indicates ERROR, and the wrapper has control of the AHB data bus.
nWAIT	Not wait	1	ANDed with **TestClk**, so that the core state can only change during the data access cycle.
nRESET	Not reset	0	-

The test data output multiplexor is found in the A7TWrapCtrl block, but is controlled by the test outputs of this block. It is used to select between:

- core data output during ST_DATA_OUT
- core address output during ST_ADDR_OUT
- core status outputs during ST_STAT_OUT.

The selected output is driven onto the **HRDATAS** output data bus.

Table 3-7 shows the bit positions of the status output signals when driven on the data bus.

Table 3-7 ARM7TDMI status bit positions

Signal	Description	Bit position	Comment
BUSDIS	Bus disable	31	-
SCREG[3:0]	Scan chain register	30:27	These signals are not important to the normal functioning of the core, but are included in this test vector to give a slight improvement in fault coverage during scan and debug testing.
HIGHZ	HIGHZ instruction in TAP controller	26	-
nTDOEN	Not **TDO** enable	25	-

Table 3-7 ARM7TDMI status bit positions (continued)

Signal	Description	Bit position	Comment
DBGRQ1	Internal debug request	24	-
RANGEOUT0	ICEbreaker Rangeout0	23	-
RANGEOUT1	ICEbreaker Rangeout1	22	-
COMMRX	Communications channel receive	21	-
COMMTX	Communications channel transmit	20	-
DBGACK	Debug acknowledge	19	-
TDO	Test data out	18	This value is often tristate (as indicated by **nTDOEN**), so is usually masked out.
nENOUT	Not enable output	17	**nENOUT** is only valid during the data access cycle, so **TestClk** is used to clock a register that captures the correct state.
nENOUTI	Not enable output	16	**nENOUTI** is only valid during the data access cycle, so **TestClk** is used to clock a register that captures the correct state.
TBIT	Thumb state	15	-
nCPI	Not coprocessor instruction	14	-
nM[4:0]	Not processor mode	13:9	-
nTRANS	Not memory translate	8	-
nEXEC	Not executed	7	-
LOCK	Locked operation	6	-
MAS[1:0]	Memory access size	5:4	-
nOPC	Not opcode fetch	3	-
nRW	Not read/write	2	-
nMREQ	Not memory request	1	-
SEQ	Sequential address	0	-

This test interface block can be removed if not required, by removing the A7TWrapTest block from the A7TWrap top level wrapper HDL file. It is then necessary to tie the outputs that were originally generated from this block to fixed values, and these are described in the A7TWrap HDL code.

Removing this block means that the test inputs to the A7TWrapCtrl block is static, allowing the test multiplexors to be removed during synthesis, or manually removed from the HDL code.

The AHB slave outputs are only used during TIC testing mode. **HREADYOUTS** is always driven HIGH, as the wrapper never generates wait states. **HRESPS** is always driven to OKAY, as the wrapper never asserts split, retry or error responses.

HRDATAS is generated according to the current test control signal outputs, and is driven to either **DOUT** from the core, **TestData** from the test block (which is comprised of the core control outputs), or LOW.

3.3.8 A7TWrapCtrl

This block contains the test wrapper control multiplexor used during TIC testing of the core. A simplified diagram of the A7TWrapCtrl block is shown in Figure 3-5.

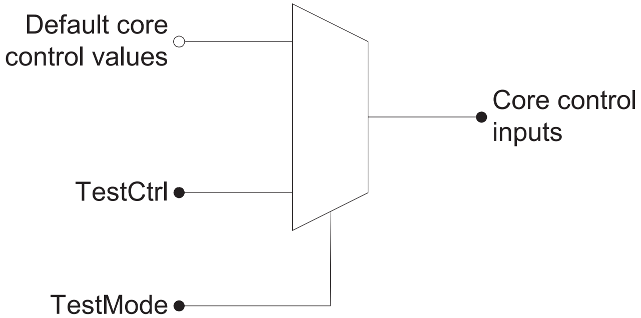

Figure 3-5 A7TWrapCtrl block system diagram

This block is only used during test mode when the wrapper is acting as an AHB slave, and drives the control inputs of the core with the TIC test data. It is separated from the main test block (A7TWrapTest) to allow for easier removal of the test wrapper.

When not in test mode the control inputs are driven to their default values (either HIGH or LOW), or are driven with wrapper inputs, such as the two interrupt lines and the JTAG pins. The **AbortInt** signal is generated in the A7WrapMaster block.

If the test wrapper is removed, then this multiplexor is optimized out during synthesis, and the outputs driven with their default values. It is also possible to remove this block if the test wrapper is not used. The default connections that the outputs must be tied to are shown in the A7TWrap HDL file.

BUSEN, **DBE**, and **nENIN** are all set to constant values, because they do not have to be controlled during normal system use, or during core TIC testing.

3.4 Non-standard design practices

The following non-standard design practices are described:

- *Clock gating*
- *Transparent latches*.

Note

The following applies only to the ARM7TDMI AHB wrapper, and not the ARM7TDMI-S AHB wrapper.

3.4.1 Clock gating

A clock inverter is instantiated within A7TWrap (instance name *nHCLKgen*) to provide **nHCLK** (inverted **HCLK**). This signal is used in the following places:

A7TWrap	Creation of **dCLKEN**, one of the enable terms for **MCLK**.
A7TWrapTest	Creation of **TestModeF**, the other enable term for **MCLK**.

An inverted version of the clock is used so that all HDL code describes only rising-edge sequential logic. This is done because some cell libraries do not contain falling-edge registers, and also to avoid possible insertion of unwanted clock gating during synthesis.

A clock NAND gate is used within A7TWrap (instance name *MCLKgen*) to create **MCLK**, the clock for the memory interface of the ARM7TDMI.

The clock gates are described in the design block ClockInv and ClockNand. The methodology chosen within the supplied synthesis scripts synthesizes these blocks first, sets them as *dont_touch*, and then links them in when synthesizing the wrapper. This ensures that the clock gate instances have known references, and also prevents the optimization routines during later synthesis from altering the gates.

3.4.2 Transparent latches

A transparent latch is instantiated within A7TWrap (instance name *DelCLKEN*). This is use to create **dCLKEN**, one of the enable terms for **MCLK**. The latch operation is described in the design block LATS (transparent latch with asynchronous set). The synthesis methodology for this block is as for the clock gates described above.

Chapter 4
ARM920T AHB Wrapper

This chapter describes the ARM920T and ARM922T processor core wrapper that you can use with an AHB-based EASY system. It contains the following sections:

4.1 About the ARM920T AHB wrapper

The ARM920T AHB wrapper interfaces between the ARM920T (or ARM922T) core and the AHB bus. The wrapper itself sits alongside the ARM920T core, intercepting the ASB bus from the core. An example higher-level module is also included. This shows how the core and wrapper can be connected, and is used by the synthesis scripts provided to allow the wrapper to be synthesized alongside a timing file for the core.

The wrapper intercepts both the ASB master bus (used during normal operation of the core) and the ASB slave bus (used during testing of the ARM920T core when the *Test Interface Controller* (TIC) is the current AHB master). Figure 4-1 shows a top-level block diagram of the ARM920T AHB wrapper.

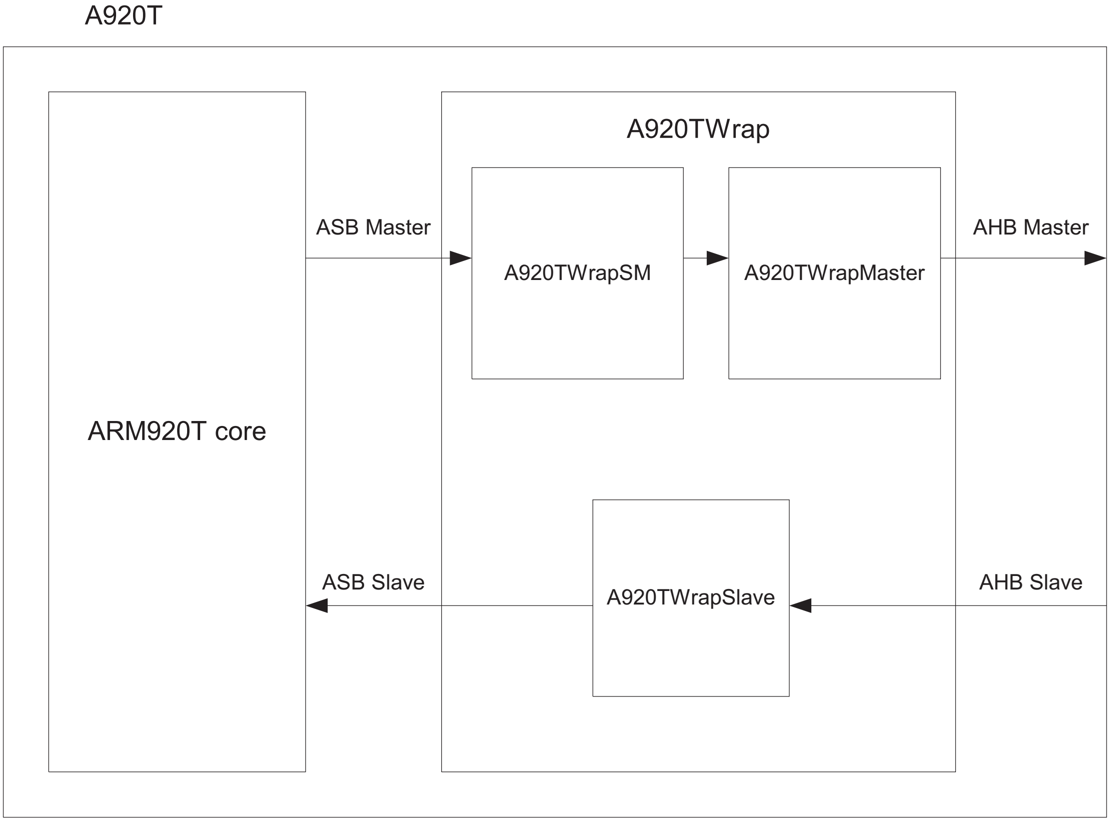

Figure 4-1 ARM920T AHB wrapper block diagram

4.2 Signal interface

This section describes the signal interface of A920TWrap. It does not describe the ARM920T. Only those signals from the ARM920T core that are used by the AHB wrapper are described. All other core signals (for example, coprocessor interface and interrupt inputs) must be connected to the external system in the same way as for an ASB-native design. See the *ARM920T Datasheet* for the relevant information.

Note

The exception is **BCLK**, which must be connected directly to the AHB system clock **HCLK**.

Table 4-1 describes the signals used by the ARM920T AHB wrapper.

Table 4-1 ARM920T AHB signal descriptions

Signal	Direction	Description
System inputs		
HCLK	Input	Bus clock. This clock times all bus transfers. All signal timings are related to the rising edge of **HCLK**.
HRESETn	Input	Reset. The bus reset signal is active LOW and is used to reset the system and the bus. This is the only active LOW AHB signal.
Master inputs		
HRDATAM[31:0]	Input	Read data bus. Used to transfer data to the ARM920T in master mode.
HREADYM	Input	Transfer done. When HIGH the **HREADYM** signal indicates that a transfer has finished on the bus. This signal can be driven LOW to extend a transfer.
HRESPM[1:0]	Input	Transfer response. Indicates an OKAY, ERROR, RETRY, or SPLIT response.
HGRANTM	Input	Bus grant. Indicates that the ARM920T is currently the highest priority master. Ownership of the address/control signals changes at the end of a transfer when **HREADYM** is HIGH, so a master gains access to the bus when both **HREADYM** and **HGRANTM** are HIGH.
Master outputs		
HADDRM[31:0]	Output	This is the 32-bit system address bus.
HTRANSM[1:0]	Output	Transfer type. This signal indicates the type of the current transfer, which can be NONSEQUENTIAL, SEQUENTIAL, IDLE, or BUSY.

Table 4-1 ARM920T AHB signal descriptions (continued)

Signal	Direction	Description
HWRITEM	Output	Transfer direction. When HIGH, this signal indicates a write transfer and when LOW a read transfer.
HSIZEM[2:0]	Output	Transfer size. Indicates the size of the transfer, which can be byte (8-bit), halfword (16-bit), or word (32-bit).
HBURSTM[2:0]	Output	Burst type. Indicates if the transfer forms part of a burst. The ARM920T performs incrementing bursts of type INCR, INCR4, or INCR8.
HPROTM[3:0]	Output	Protection control. These signals indicate if the transfer is an opcode fetch or data access, and if the transfer is a Supervisor mode access or User mode access.
HWDATAM[31:0]	Output	Write data bus. Used to transfer data from the ARM920T in master mode.
HBUSREQM	Output	Bus request. A signal from the wrapper to the bus arbiter that indicates that it requires the bus.
HLOCKM	Output	Locked transfer. When HIGH, this signal indicates that the master requires locked access to the bus and no other master must be granted the bus until this signal is LOW.
Slave inputs		
HADDRS[11:2]	Input	This is the 32-bit system address bus.
HTRANS1S	Input	Transfer type. This is attached to bit 1 of the AHB **HTRANS[1:0]** bus. It indicates an active (NONSEQ or SEQ) or inactive (IDLE or BUSY) transfer.
HWRITES	Input	Transfer direction. When HIGH, this signal indicates a write transfer and when LOW a read transfer.
HWDATAS[31:0]	Input	Write data bus. Used to transfer data to the ARM920T in slave mode.
HSELS	Input	This signal selects the ARM920T as slave.
HREADYS	Input	Transfer done. When HIGH, this signal indicates that a transfer has finished on the bus. This signal can be driven LOW to extend a transfer.
Slave outputs		
HRDATAS[31:0]	Output	Read data bus. Used to transfer data from the ARM920T in slave mode.
HREADYOUTS	Output	Transfer done. When HIGH, this signal indicates that a transfer to the ARM920T has finished. This signal can be driven LOW to extend a transfer.
HRESPS[1:0]	Output	Transfer response. Indicates an OKAY, ERROR, RETRY, or SPLIT response. The ARM920T always responds with OKAY.

Table 4-2 describes the signals to the ARM920T core.

Table 4-2 Signals to ARM920T core

Signal	Direction	Description
Shared master/slave signals		
BnRES	Output	Bus reset signal which is active LOW.
DIN[31:0]	Output	Data bus. For master mode this is connected to **HRDATAM**, for test mode it is connected to **HWDATAS**.
DOUT[31:0]	Input	Data bus. For master mode this is connected to **HWDATAM**, for test mode it is connected to **HRDATAS**.
Master inputs		
AOUT[31:0]	Input	This is the ASB address bus for master mode.
AREQ	Input	Bus request. Indicates that the ARM920T is requesting the bus.
ASTB	Input	Indicates that a transfer commences on the next cycle.
BURST[1:0]	Input	Burst type. Contains information regarding the type of access being performed by the core.
LOK	Input	Locked transfer. The ARM920T requires a locked bus transfer during a SWP instruction.
NCMAHB	Input	Indicates that there is another transfer in the current burst read from a noncacheable memory area.
PROT[1:0]	Input	Protection control. Indicates whether the current access is for data or opcode, and privileged or user level access.
SIZE[1:0]	Input	Transfer size. Indicates the size of the transfer, which can be byte (8-bit), halfword (16-bit), or word (32-bit).
TRAN[1:0]	Input	Transfer type. Indicates the type of the current transfer, which can be NONSEQUENTIAL, SEQUENTIAL, or ADDRESS-ONLY.
WRITEOUT	Input	Transfer direction. Indicates a write access in master mode.
Master outputs		
AGNT	Output	Bus grant. Indicates that the ARM920T has control of the ASB.
WAITIN	Output	Wait response. Indicates wait states to the core.
ERRORIN	Output	Error response. Indicates that an error has occurred during the access.

Table 4-2 Signals to ARM920T core (continued)

Signal	Direction	Description
Slave inputs		
WAITOUT	Input	Wait response. Indicates wait states during slave mode.
Slave outputs		
AIN[11:2]	Output	Slave mode address bus.
DSEL	Output	Select line for accesses to the core during slave mode.
WRITEIN	Output	Transfer direction. Indicates a write access in slave mode.

4.3 Description of the ARM920T wrapper blocks

This section contains descriptions of the following blocks:

- *A920T*
- *A920TWrap*
- *A920TWrapSM*
- *A920TWrapMaster* on page 4-12
- *A920TWrapSlave* on page 4-13.

4.3.1 A920T

This is an example top level containing instantiations of both the ARM920T core and the AHB wrapper. It is used by the provided synthesis scripts to allow synthesis of the wrapper alongside a characterized timing file of the core. All core signals not connected to the AHB wrapper (such as the coprocessor interface) are shown with example connections only. You can alter these, or use a completely new top-level in the design.

4.3.2 A920TWrap

This instantiates the master and slave components of the AHB wrapper. It also instantiates a clock gate to provide an inverted version of **HCLK** to some functions in the sub-blocks, and infers a mux to select the correct source for DIN.

4.3.3 A920TWrapSM

In master mode, this block converts the ASB accesses from the core into pseudo-AHB accesses. This pseudo-AHB format is similar to AHB-Lite. The following sections of the code are explained:

- *Special signals* on page 4-8
- *Main state machine* on page 4-8
- *Address generation* on page 4-10
- *Wait states* on page 4-11
- *Error support* on page 4-11
- *Control signaling (HTRANS, HSIZE, HBURST, HPROT)* on page 4-12.

Special signals

Two signals exist on the ARM920T to improve the performance of the AHB wrapper:

NCMAHB Indicates when there is another access (after the current one) in a burst of reads to a noncacheable region. Allows the AHB address phase of the next read to commence before the access commences on the ASB, by use of a local address incrementer.

ASTB Indicates that the current IDLE cycle on the ASB is the first part of a merged I-S cycle, That is, the control information contained within the IDLE cycle is correct for the following SEQ cycle, and so the access can, in some cases, commence on the AHB in advance.

Main state machine

The main state machine uses the state variable AddrState (which is registered to create LastAddrState). This can take the following enumerated values:

IDLE Default state. Indicates that no transfer is taking place on the AHB.

READ_START

First cycle of read transfer. A NONSEQ access is indicated on **HTRANS**.

READ_HELD

Indicates that the first cycle of a read transfer has been waited. A NONSEQ access is indicated on **HTRANS**.

READ_SEQ

Indicates sequential transfers in a burst of reads. A SEQ transfer is indicated on **HTRANS**.

LOCK_STALL

Start of SWP instruction. This state stalls the core to allow **HLOCK** to be asserted for one cycle before commencing the read transaction. An IDLE cycle is indicated on **HTRANS**.

WRITE_ASTB

A wait is indicated on the ASB but cannot yet commence on the AHB. An IDLE cycle is indicated on **HTRANS**.

NBUF_WRITE_START

First cycle of a nonbuffered write transfer. A NONSEQ access is indicated on **HTRANS**.

NBUF_WRITE_DATA

Synchronization state to allow the response to the previous write transfer to propagate to the core before commencing the next transfer. A BUSY transfer is indicated on **HTRANS** when more writes are likely to follow, else IDLE.

NBUF_WRITE_SEQ

Indicates sequential transfers in a burst of nonbuffered writes. A SEQ transfer is indicated on **HTRANS**.

BUF_WRITE_START

First cycle of a buffered write transfer. A NONSEQ access is indicated on **HTRANS**.

BUF_WRITE_SEQ

Indicates sequential transfers in a burst of buffered writes. A SEQ transfer is indicated on **HTRANS**.

Figure 4-2 on page 4-10 shows the possible state transitions. State transitions occur on the rising edge of **HCLK**. Transitions back to the same state can occur either when the access on the pseudo-AHB is waited, or during sequential accesses in a burst (BUF_WRITE_SEQ and READ_SEQ only). Those states which cannot transition back to themselves (WRITE_ASTB, READ_START, LOCK_STALL) provide a single cycle to allow synchronization between events on the ASB and the AHB.

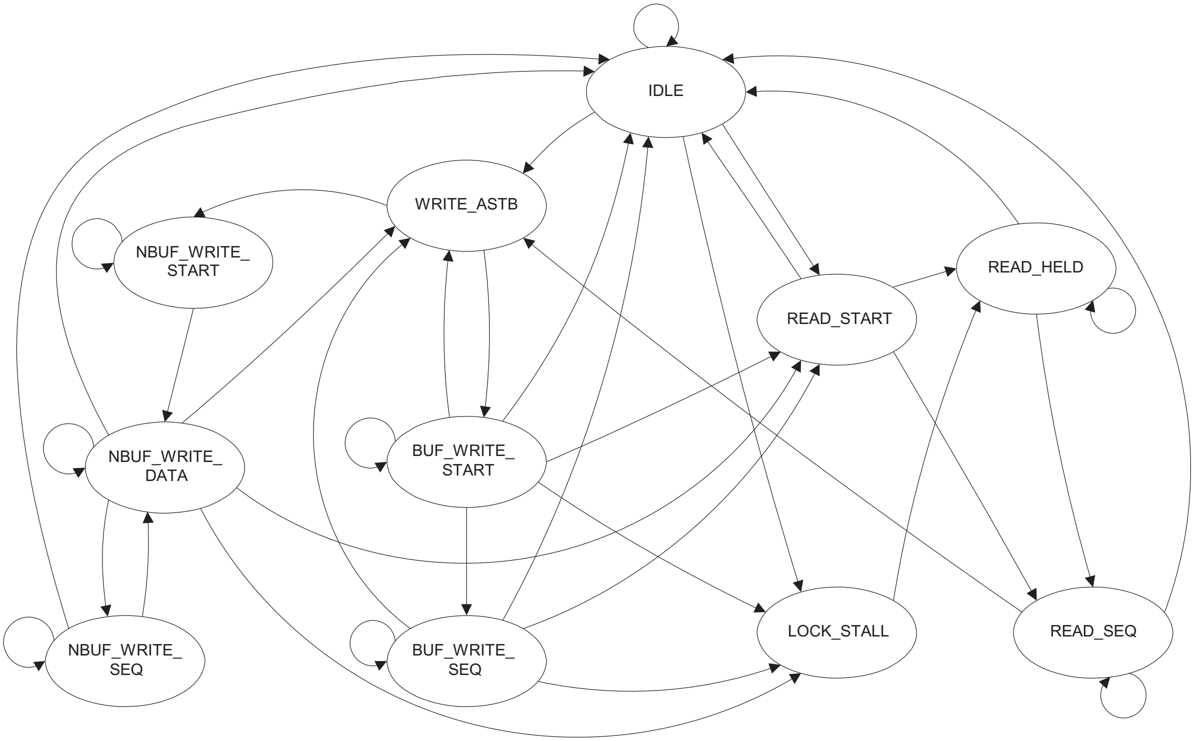

Figure 4-2 ARM920T AHB wrapper main state machine

Address generation

There are three internal sources of address:

- direct from the ARM920T core
- a registered version of the core address
- a locally incremented address created within A920TWrapSM.

The direct address is used when the core access and the AHB access occur in the same clock cycle. The registered version of the core address is used when the AHB access is waited, but the core has been allowed to step on to determine the next access to be performed. The locally incremented address is used when the AHB access commences before the core access (for example, when the **NCMAHB** signal indicates more accesses in the current burst of reads).

Wait states

The inclusion of the **NCMAHB** and **ASTB** signals has enabled the ARM920T AHB wrapper to be designed in such a way that a minimum number of extra wait states are added by the conversion from ASB to AHB. Wait states on the AHB (indicated by **HREADY** being LOW) are propagated to the ARM920T only when the core is attempting to access the bus. Because of this, a string of IDLE cycles on the ASB is not waited. Additional wait states (where one wait state equates to a single cycle of **HCLK**) are added as follows:

- +1 wait state on each nonbuffered write (either a single access, or each beat in a burst). This allows for the AHB ERROR response to propagate back to the ASB.
- +1 wait state on the read at the start of a SWP instruction. This allows for **HLOCK** to be asserted for a cycle before the AHB access commences.

Error support

The ARM920T AHB wrapper includes support for the ERROR response on **HRESP**. The ERROR response is only valid for the following accesses:

- read and write accesses to NCNB address regions
- read accesses to NCB address regions.

For all other access types, the ERROR response is blanked within the AHB wrapper. This is because, due to the pipelining within the AHB wrapper, an error response to certain CB accesses would otherwise appear on the ASB in response to the next access.

Note

Page table walks are a special case, and ERROR response is not supported for these accesses (either by the ARM920T core, or by the AHB wrapper).

Control signaling (HTRANS, HSIZE, HBURST, HPROT)

Table 4-3 shows the control signal values used by the ARM920T AHB wrapper.

Table 4-3 Control signal values

Signal	Value
HTRANS	00: IDLE 01: BUSY 10: NONSEQUENTIAL 11: SEQUENTIAL
HSIZE	00: BYTE 01: HALFWORD 10: WORD
HBURST	001: INCR 011: INCR4 101: INCR8
HPROT	bit 3: 1 = cacheable, 0 = not cacheable (only valid for reads, 0 for all writes) bit 2: 1 = bufferable, 0 = not bufferable (only valid for writes, 0 for all reads) bit 1: 1 = privileged access, 0 = user access bit 0: 1 = data access, 0 = opcode fetch.

Note

The cached status of page table walks cannot be correctly determined outside of the core, and **HPROT[3]** is set to 1 for these accesses.

4.3.4 A920TWrapMaster

In master mode, this block converts the pseudo-AHB accesses from A920TWrapSM into true AHB by adding support for split and retry responses, and also for **HGRANT**. A split/retry response, or loss of **HGRANT**, causes the affected access to be placed into a holding register for reconstruction when the access can recommence on the AHB. During this time, the pseudo-AHB access is simply waited using **MREADY**.

The holding registers store the current transfer on the pseudo-AHB on each valid cycle (qualified by **MREADY**). The AHB outputs are multiplexed based on the signal **HoldSel**. This is a registered signal, which is synchronously set and cleared by **HoldSet** and **HoldClr** respectively:

- **HoldSet** is activated when a SPLIT or RETRY response is received during the data phase of an access to the core, or when ownership of the AHB bus is lost attempting an active transfer (that is, NONSEQ or SEQ).
- **HoldClr** is activated when **HoldSel** is active, the AHB wrapper has ownership of the bus and any SPLIT or RETRY response has completed.

When reconstructing a burst after the use of the holding registers, to meet AHB protocol, the wrapper forces assertion of **INCR** on **HBURST** when the number of transfers remaining is unknown (for example, if only three transfers remain from an INCR8 burst). In cases where an entire INCR4 or INCR8 burst remains to be transferred (for example, when the core was not initially granted the AHB bus and so must wait in the holding state until **HGRANT** is asserted), the burst size information is retained.

4.3.5 A920TWrapSlave

In slave mode, this block converts incoming AHB accesses into ASB accesses to the ARM920T core. Because slave testing of the core is only designed for low-speed operation, retiming registers are not included in these paths. This allows for no extra wait states to be added into the test procedure, and speeds up production testing.

Two transparent latches are instantiated (using the supplied LATR component) to hold the **WRITEIN** and **DSEL** inputs to the core stable during the LOW phase of **HCLK**.

Note

The connections between the AHB and ASB for slave testing result in some paths having only a single phase of **HCLK** (as opposed to a full cycle) in which to complete. Therefore slave testing of the ARM920T cannot be carried out at the same maximum frequency as can be achieved when the core is the AHB bus master. It is recommended that slave testing be limited to one-quarter of the maximum attainable **HCLK** frequency.

4.4 Non-standard design practices

The following non-standard design practices are described:

- *Clock gating*
- *Transparent latches*.

4.4.1 Clock gating

A clock inverter is instantiated within A920TWrap (instance name *nHCLKgen*) to provide **nHCLK** (inverted **HCLK**). This signal is used in two places:

A920TWrapSM Resynchronization of **HRESETn** to deassert **BnRES** on the falling edge of **HCLK**.

A920TWrapSlave Creation of **AGNT** off falling edge of **HCLK**.

An inverted version of the clock is used so that all HDL code describes only rising-edge sequential logic. This is done because some cell libraries do not contain falling-edge registers, and also to avoid possible insertion of unwanted clock gating during synthesis.

The clock inverter is described in the design block ClockInv. The methodology chosen within the supplied synthesis scripts synthesizes this block first, sets it as *dont_touch*, and then links it in when synthesizing the wrapper. This ensures that the clock gate instance has a known reference, and also prevents the optimization routines during later synthesis from altering the gate.

4.4.2 Transparent latches

Two transparent latches are instantiated within A920TWrapSlave (instance names *uLATRwrite* and *uLATRdsel*). These are required to delay the core inputs **WRITEIN** and **DSEL** respectively during TIC testing. A simple register cannot be used because the incoming signals arrive too late.

The latch operation is described in the design block LATR (transparent latch with active-low asynchronous reset). The synthesis methodology for this block is the same as for ClockInv described above.

4.5 Programmer's model exceptions

The Drain Write Buffer command is available inside CP15 register 7 using the following ARM instruction:

```
MCR p15, 0, Rd, c7, c10, 4
```

The expected operation is that any entries in the write buffer will be flushed, and that core execution will halt until the flush has completed. The ARM920T AHB wrapper cannot know when this command is being used, and treats all buffered writes the same. The write operation is accepted from the core, and the core then allowed to continue to the next operation, even if the data phase of the write access on the AHB is delayed (wait states, loss of grant, SPLIT or RETRY response). This means that fully cached operations following the drain command can execute before the write has taken effect in the outside world.

If core operation must be halted after a drain command, add a further instruction which will cause a bus access (such as a load from non-cacheable memory, or a store to non-bufferable memory). This extra access will not complete until the write buffer flush has completed, and will store the core.

Chapter 5
ARM940T AHB Wrapper

This chapter describes the ARM940T processor core wrapper that you can use with an AHB-based EASY system. It contains the following sections:

5.1 About the ARM940T AHB wrapper

The ARM940T AHB wrapper interfaces between the ARM940T core and the AHB bus. The wrapper itself sits alongside the ARM940T core, intercepting the ASB bus from the core. An example higher-level module is also included. This shows how the core and wrapper can be connected, and is used by the synthesis scripts provided to allow the wrapper to be synthesized alongside a timing file for the core.

The wrapper intercepts both the ASB master bus (used during normal operation of the core) and the ASB slave bus (used during testing of the ARM940T core when the *Test Interface Controller* (TIC) is the current AHB master). Figure 5-1 shows a top-level block diagram of the ARM940T AHB wrapper.

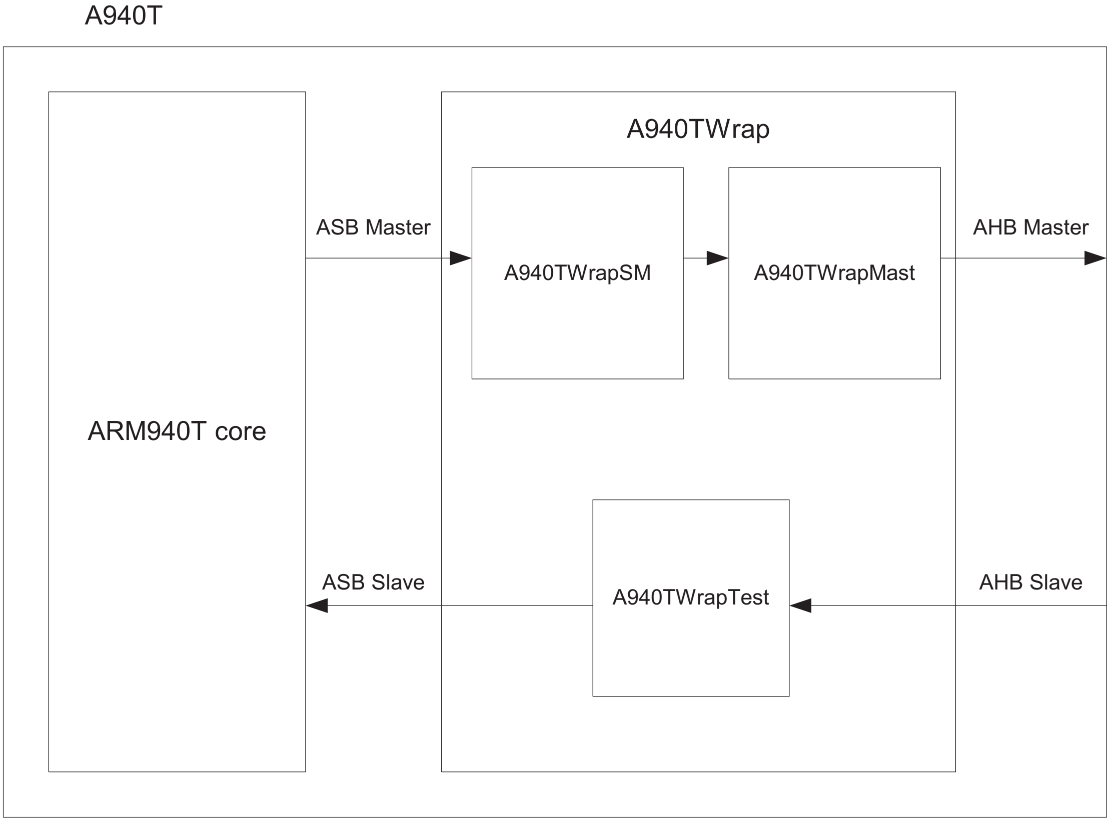

Figure 5-1 ARM940T AHB wrapper block diagram

5.2 Signal interface

This section describes the signal interface of A940TWrap. It does not describe the ARM940T. Only those signals from the ARM940T core that are used by the AHB wrapper are described. All other core signals (for example, coprocessor interface and interrupt inputs) must be connected to the external system in the same way as for an ASB-native design. See the *ARM940T Datasheet* for the relevant information.

Note

The exception is **BCLK**, which must be connected directly to the AHB system clock **HCLK**.

Table 5-1 describes the signals used by the ARM940T AHB wrapper.

Table 5-1 ARM940T AHB signal descriptions

Signal	Direction	Description
System inputs		
HCLK	Input	Bus clock. This clock times all bus transfers. All signal timings are related to the rising edge of **HCLK**.
HRESETn	Input	Reset. The bus reset signal is active LOW and is used to reset the system and the bus. This is the only active LOW AHB signal.
Master inputs		
HRDATAM[31:0]	Input	Read data bus. Used to transfer data to the ARM940T in master mode.
HREADYM	Input	Transfer done. When HIGH, this signal indicates that a transfer has finished on the bus. This signal can be driven LOW to extend a transfer.
HRESPM[1:0]	Input	Transfer response. Indicates an OKAY, ERROR, RETRY, or SPLIT response.
HGRANTM	Input	Bus grant. This signal indicates that the ARM940T is currently the highest priority master. Ownership of the address/control signals changes at the end of a transfer when **HREADYM** is HIGH, so a master gains access to the bus when both **HREADYM** and **HGRANTM** are HIGH.
Master outputs		
HADDRM[31:0]	Output	This is the 32-bit system address bus.
HTRANSM[1:0]	Output	Transfer type. This signal indicates the type of the current transfer, which can be NONSEQUENTIAL, SEQUENTIAL, IDLE, or BUSY.

Table 5-1 ARM940T AHB signal descriptions (continued)

Signal	Direction	Description
HWRITEM	Output	Transfer direction. When HIGH, this signal indicates a write transfer and when LOW a read transfer.
HSIZEM[2:0]	Output	Transfer size. Indicates the size of the transfer, which can be byte (8-bit), halfword (16-bit), or word (32-bit).
HBURSTM[2:0]	Output	Burst type. Indicates if the transfer forms part of a burst. The ARM940T performs incrementing bursts of type INCR, INCR4, or INCR8.
HPROTM[3:0]	Output	Protection control. These signals indicate if the transfer is an opcode fetch or data access, and if the transfer is a Supervisor mode access or User mode access.
HWDATAM[31:0]	Output	Write data bus. Used to transfer data from the ARM940T in master mode.
HBUSREQM	Output	Bus request. A signal from the wrapper to the bus arbiter which indicates that it requires the bus.
HLOCKM	Output	Locked transfers. When HIGH, this signal indicates that the master requires locked access to the bus and no other master must be granted the bus until this signal is LOW.
Slave inputs		
HADDRS[11:2]	Input	This is the 32-bit system address bus.
HTRANS1S	Input	Transfer type. This is attached to bit 1 of the AHB **HTRANS[1:0]** bus. It indicates an active (NONSEQ or SEQ) or inactive (IDLE or BUSY) transfer.
HWRITES	Input	Transfer direction. When HIGH, this signal indicates a write transfer and when LOW a read transfer.
HWDATAS[31:0]	Input	Write data bus. Used to transfer data to the ARM940T in slave mode.
HSELS	Input	This signal selects the ARM940T as slave.
HREADYS	Input	Transfer done. When HIGH, this signal indicates that a transfer has finished on the bus. This signal can be driven LOW to extend a transfer.
Slave outputs		
HRDATAS[31:0]	Output	Read data bus. Used to transfer data from the ARM940T in slave mode.
HREADYOUTS	Output	Transfer done. When HIGH the **HREADYOUTS** signal indicates that a transfer to the ARM940T has finished. This signal can be driven LOW to extend a transfer.
HRESPS[1:0]	Output	Transfer response. Indicates an OKAY, ERROR, RETRY, or SPLIT response. The ARM940T always responds with OKAY.

Table 5-2 describes the signals to the ARM940T core.

Table 5-2 Signals to ARM940T core

Signal	Direction	Description
Shared master/slave signals		
BnRES	Output	Bus reset signal which is active LOW.
BD[31:0]	Input/output	Data bus. For master mode this is connected to **HRDATAM** and **HWDATAM**, for test mode it is connected to **HRDATAS** and **HWDATAS**.
BWRITE	Input/output	Transfer direction. This is derived from **HWRITE**. When HIGH this signal indicates a write transfer and when LOW a read transfer. During test this signal is driven by the wrapper.
BERROR	Input/output	Error response. For master mode, this provides the response from the slave (DONE, ERROR, or WAIT). For slave mode, the ARM940T always responds with DONE.
BLAST	Input/output	Last response. For master mode, this provides the response from the slave (DONE, ERROR, or WAIT). For slave mode, the ARM940T always responds with DONE.
BWAIT	Input/output	Wait response. For master mode, this provides the response from the slave (DONE, ERROR, or WAIT). For slave mode, the ARM940T always responds with DONE.
Master inputs		
BA[31:0]	Input	This is the ASB address bus for master mode.
AREQ	Input	Bus request. Indicates that the ARM940T is requesting the bus.
BURST[1:0]	Input	Burst type. Contains information regarding the type of access being performed by the core.
BLOK	Input	Locked transfer. The ARM940T requires a locked bus transfer during a SWP instruction.
BPROT[1:0]	Input	Protection control. Indicates whether the current access is for data or opcode, and privileged or user level access.
BSIZE[1:0]	Input	Transfer size. Indicates the size of the transfer, which can be byte (8-bit), halfword (16-bit), or word (32-bit).
BTRAN[1:0]	Input	Transfer type. Indicates the type of the current transfer, which can be NONSEQUENTIAL, SEQUENTIAL, or ADDRESS-ONLY.
Master outputs		
AGNT	Output	Bus grant. Indicates that the ARM940T has control of the ASB.
Slave outputs		
DSEL	Output	Slave select. Select line for accesses to the core during slave mode.

5.3 Description of the ARM940T wrapper blocks

This section contains descriptions of the following blocks:

- *A940T*
- *A940TWrap*
- *A940TWrapSM*
- *A940TWrapMast* on page 5-11
- *A940TWrapTest* on page 5-12.

5.3.1 A940T

This is an example top level containing instantiations of both the ARM940T core and the AHB wrapper. It is used by the provided synthesis scripts to allow synthesis of the wrapper alongside a characterized timing file of the core. All core signals not connected to the AHB wrapper (such as the coprocessor interface) are shown with example connections only. You can alter these, or use a completely new top-level in the design.

5.3.2 A940TWrap

This instantiates the master and slave components of the AHB wrapper. It also instantiates a clock gate to provide an inverted version of **HCLK** to some functions in the sub-blocks.

5.3.3 A940TWrapSM

In master mode, this block converts the ASB accesses from the core into pseudo-AHB accesses. This pseudo-AHB format is similar to AHB-Lite. The following sections of the code are explained:

- *Main state machine*
- *Address generation* on page 5-9
- *Wait states* on page 5-10
- *Error support* on page 5-10
- *Control signaling (HTRANSM, HSIZEM, HBURSTM, HPROTM)* on page 5-11.

Main state machine

The main state machine uses the state variable CurrentState (which is registered to create LastState). This can take the following enumerated values:

IDLE Default state. Indicates that no transfer is taking place on the AHB. An IDLE cycle is indicated on **HTRANSM**.

LOCK_STALL

Start of SWP instruction. This state stalls the core to allow **HLOCKM** to be asserted for one cycle before commencing the read transaction. An IDLE cycle is indicated on **HTRANSM**.

READ_START

First cycle of a read transfer. A NONSEQ access is indicated on **HTRANSM**.

READ_BURST

Second and subsequent cycles of a 4-beat cache linefill. A SEQ access is indicated on **HTRANSM**.

READ_SEQ

Indicates sequential accesses in an undefined length burst of reads. A SEQ access is indicated on **HTRANSM**.

READ_BUSY

Indicates that the current undefined length burst of reads can continue. A BUSY cycle is indicated on **HTRANSM**.

READ_STALL

Indicates that the current undefined length burst of reads has been stalled but is to continue. A SEQ access is indicated on **HTRANSM**.

WRITE_START

First cycle of a write transfer. A NONSEQ access is indicated on **HTRANSM**.

WRITE_BURST

Second and subsequent cycles of a 4-beat write-buffer flush. A SEQ access is indicated on **HTRANSM**.

WRITE_BUSY

Indicates that the current undefined length burst of writes is to continue after the current access has completed. A BUSY cycle is indicated on **HTRANSM**.

WRITE_SEQ

Indicates sequential accesses in an undefined length burst of writes. A SEQ access is indicated on **HTRANSM**.

LAST_DATA

Allows the last access of a burst transfer to complete. An IDLE cycle is indicated on **HTRANSM**.

Figure 5-2 shows the possible state transitions. State transitions occur on the rising edge of **HCLK**. Transitions back to the same state can occur either when the access on the pseudo-AHB is waited, or during sequential accesses in a burst (WRITE_BURST and READ_BURST only). Those states that cannot transition back to themselves provide a single cycle to allow synchronization between events on the ASB and the AHB.

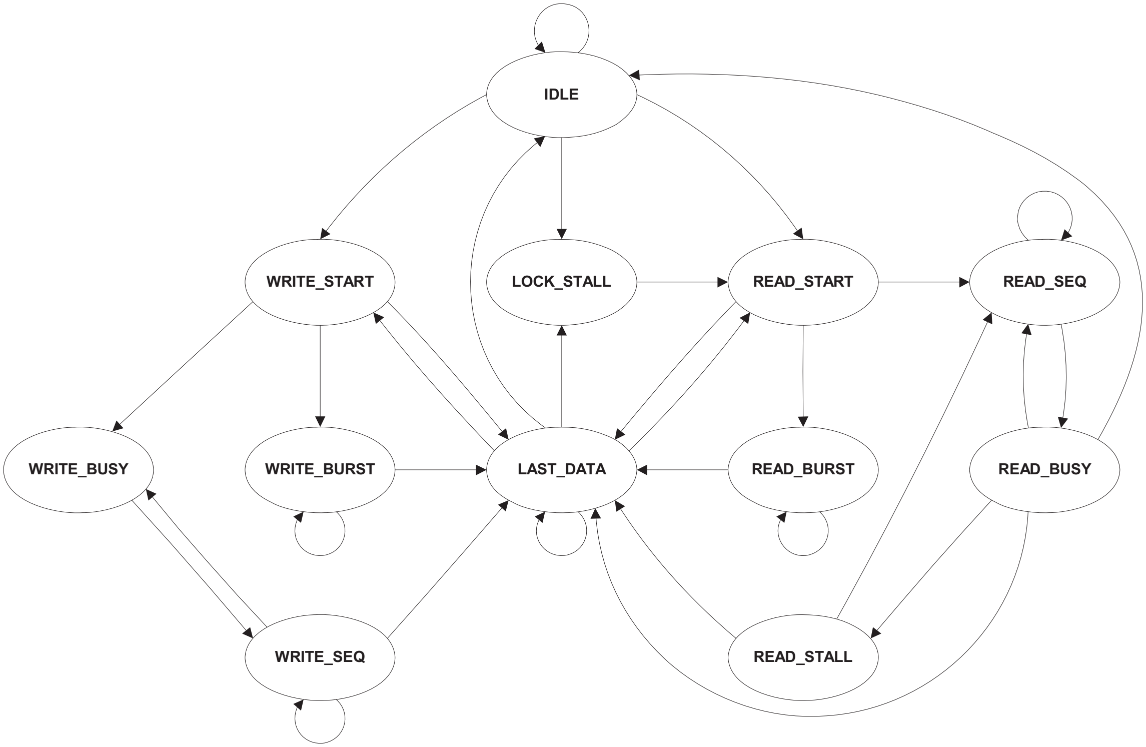

Figure 5-2 ARM940T AHB wrapper main state machine

Address generation

There are two internal sources of address:

- a registered version of the core address
- a locally incremented address created within A940TWrapSM.

The incremented address is used when sequential transfers on the AHB occur in the same cycle as on the ASB. At all other times, the registered version is used.

Wait states

Wait states on the AHB (indicated by **HREADYM** being LOW) are propagated to the ARM940T only when the core is attempting to access the bus. Because of this, a string of IDLE cycles on the ASB is not waited. Additional wait states (where one wait state equates to a single cycle of **HCLK**) are added as follows:

- +1 at the start of each 4-beat cacheable or bufferable burst.
- +1 for each beat of a burst write of undefined length. These are masked by wait states from the slave being accessed except for the final wait state at the end of the burst.
- +1 for each beat of a burst read of undefined length. These are masked by wait states from the slave being accessed except for the initial wait state at the beginning of the burst.
- +3 for a SWP instruction (comprising +2 for the read, +1 for the write).

Error support

The ARM940T AHB wrapper includes support for the ERROR response on **HRESPM**. The ERROR response is only valid for the following accesses:

- read and write accesses to NCNB address regions
- read accesses to NCB address regions.

Control signaling (HTRANSM, HSIZEM, HBURSTM, HPROTM)

Table 5-3 shows the control signal values used by the ARM940T AHB wrapper.

Table 5-3 Control signal values

Signal	Value
HTRANSM	00: IDLE 01: BUSY 10: NONSEQUENTIAL 11: SEQUENTIAL
HSIZEM	00: BYTE 01: HALFWORD 10: WORD

Table 5-3 Control signal values (continued)

Signal	Value
HBURSTM	001: INCR 011: INCR4
HPROTM	bit 3: 1 = cacheable, 0 = not cacheable (only valid for cache linefills, 0 for all other accesses) bit 2: 1 = bufferable, 0 = not bufferable (only valid for 4-beat write buffer flushes, 0 for all other accesses) bit 1: 1 = privileged access, 0 = user access bit 0: 1 = data access, 0 = opcode fetch

——— **Note** ———

It is not possible within the AHB wrapper to fully determine the cacheable/bufferable properties of the memory area being accessed. Instead, the cacheable bit is set only when a 4-beat read is in progress (cache linefill), and the bufferable bit is set only when a 4-beat write is in progress (write buffer flush). Because of this, write buffer flushes which occur when there are less than three items in the buffer are not indicated as bufferable. Also, the lower bits of **HPROTM** (privileged/user and opcode/data) only become valid very late in the clock cycle. This is because they are derived through combinational logic from the **BPROT** outputs of the ARM940T core, that are themselves set after the falling edge of **BCLK**.

5.3.4 A940TWrapMast

In master mode, this block converts the pseudo-AHB accesses from A940TWrapSM into true AHB by adding support for split and retry responses, and also for **HGRANTM**. A split/retry response, or loss of **HGRANTM**, causes the affected access to be placed into a holding register for reconstruction when the access can recommence on the AHB. During this time, the pseudo-AHB access is simply waited using **MREADY**.

The holding registers store the current transfer on the pseudo-AHB on each valid cycle (qualified by **MREADY**). The AHB outputs are multiplexed based on the signal **HoldSel**. This is a registered signal, which is synchronously set and cleared by **HoldSet** and **HoldClr** respectively:

- **HoldSet** is activated when a SPLIT or RETRY response is received during the data phase of an access to the core, or when ownership of the AHB bus is lost attempting an active transfer (that is, NONSEQ or SEQ).

- **HoldClr** is activated when **HoldSel** is active, the AHB wrapper has ownership of the bus and any SPLIT or RETRY response has completed.

When reconstructing a burst after the use of the holding registers, to meet AHB protocol, the wrapper forces assertion of **INCR** on **HBURSTM** when the number of transfers remaining is unknown (for example, if only three transfers remain from an INCR4 burst). In cases where an entire INCR4 burst remains to be transferred (for example, when the core was not initially granted the AHB bus and so must wait in the holding state until **HGRANTM** is asserted), the burst size information is retained.

5.3.5 A940TWrapTest

The slave state machine uses the state variable NextState (which is registered to create CurrentState). This can take the following enumerated values:

MASTER_MODE

Default state, no TIC testing is taking place. The ARM940T core is granted the ASB bus through **AGNT**. In all other states, the core is degranted.

TEST_RESET

Entered on receiving the first access to the slave port. A counter keeps the state machine in TEST_RESET for eight cycles, during which time the ARM940T core is held in reset through **BnRES**. The AHB is waited through **HREADYOUTS**.

LAST_RESET

End of reset sequence. **HREADYOUTS** is deasserted to allow the first test access to continue.

Note

All ARM940T TIC vectors start with a write.

TEST_WRITE

A write access is occurring.

TURNAROUND

A turnaround cycle is required between a write access and a read access. The AHB is waited while this occurs.

TEST_READ

A read access is occurring.

TEST_IDLE1, 2, 3

These three states allow for a brief pause in the stream of test accesses to the core to occur without the core being regranted the ASB. If a read or write to the core is indicated during these states, the relevant state is entered and **AGNT** is not reasserted.

EXIT_TEST

Final test state before returning to MASTER_MODE.

Figure 5-3 on page 5-14 shows the possible state transitions. State transitions occur on the rising edge of **HCLK**. Transitions back to the same state occur when the same type of access is to be performed again (either read, write, or no access).

Note

The ARM940T core always responds to slave accesses with zero wait states on the ASB.

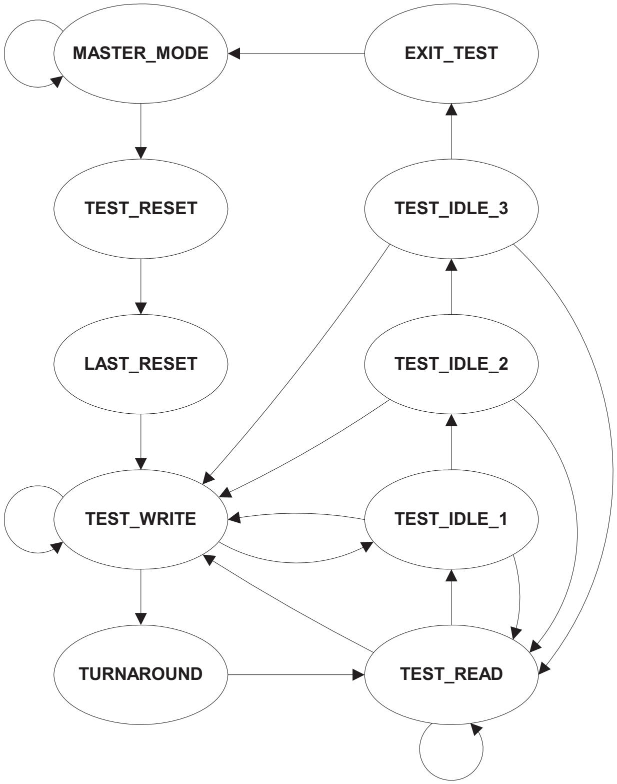

Figure 5-3 ARM940T slave state machine

5.4 Non-standard design practices

The following non-standard design practices are described:

- *Clock gating*
- *Transparent latches* on page 5-16
- *Tristate drivers* on page 5-16.

5.4.1 Clock gating

A clock inverter is instantiated within A940TWrap (instance name *uHCLKn*) to provide **HCLKn** (inverted **HCLK**). This signal is used in the following places within A940TWrapTest:

- Resynchronization of **HRESETn** to de-assert **BnRES** on the falling edge of **HCLK**.
- Creation of **AGNTarm** off falling edge of **HCLK**.
- Delaying **AGNTarm** to create DelCoreGrant, which is used in determining when to drive **BWRITE**.
- Sampling **BD** onto **HRDATAS**.

An inverted version of the clock is used so that all HDL code describes only rising-edge sequential logic. This is done because some cell libraries do not contain falling-edge registers, and also to avoid possible insertion of unwanted clock gating during synthesis.

A clock OR gate is instantiated within A940TWrapSM (instance name *uBWELEnable*) to provide **nBWELEnable**. This signal is the tristate enable for the **BWAIT**, **BERROR**, and **BLAST** outputs.

Note

During clock tree insertion, a clock stop point might have to be set on this gate to prevent the clock tree from propagating through and onto the tristate buses.

The clock gates are described in the design block ClockInv and ClockOr. The methodology chosen within the supplied synthesis scripts synthesizes these blocks first, sets them as *dont_touch*, and then links them in when synthesizing the wrapper. This ensures that the clock gate instances have known references, and also prevents the optimization routines during later synthesis from altering the gates.

5.4.2 Transparent latches

A transparent latch is instantiated within A940TWrapSM (instance name *iTRANLAT*). This is required to hold a local copy of the value from **BTRAN[1]** constant during the high phase of the clock when the **BTRAN** bus is tristate.

The latch operation is described in the design block LAT (transparent latch with no set or reset). The synthesis methodology for this block is as for the clock gates described above.

5.4.3 Tristate drivers

Tristate drivers are used within the wrapper:

- in A940TWrapSM, tristate drivers can be found for **BD**, **BWAIT**, **BLAST**, and **BERROR**
- in A940TWrapTest, tristate drivers can be found for **BWRITE** and **BD**.

Most ASIC design flows require Buskeepers to be added to any signals that might be tristated. For the A940T wrapper, the following signals must have Buskeepers added:

- **BA**
- **BD**
- **BERROR**
- **BLAST**
- **BLOK**
- **BPROT**
- **BSIZE**
- **BTRAN**
- **BWAIT**
- **BWRITE**
- **BURST**.

Index