L220 MBIST Controller

Revision: r1p7

Technical Reference Manual

ARM®

ARM DDI 0330F

L220 MBIST Controller
Technical Reference Manual

Release Information

Change history

Date	Issue	Confidentiality	Change
11 Aug 2004	A	Non-Confidential	First release.
23 Feb 2005	B	Non-Confidential	Correction to description of MBISTRESULT[2:0] on page 2-16. Updated to r1p2.
02 Aug 2005	C	Non-Confidential	Updated to r1p3.
13 Dec 2005	D	Non-Confidential	Updated to r1p4.
04 May 2006	E	Non-Confidential	Updated to r1p5.
08 Sep 2006	F	Non-Confidential	Updated to r1p7.

Proprietary Notice

Confidentiality Status

This document is Non-Confidential. The right to use, copy and disclose this document may be subject to license restrictions in accordance with the terms of the agreement entered into by ARM and the party that ARM delivered this document to.

Product Status

The information in this document is final, that is for a developed product.

Web Address

http://www.arm.com

Contents
L220 MBIST Controller Technical Reference Manual

List of Tables
L220 MBIST Controller Technical Reference Manual

List of Figures
L220 MBIST Controller Technical Reference Manual

Preface

This preface introduces the *L220 MBIST Controller Technical Reference Manual*. It contains the following sections:

About this manual

This is the *Technical Reference Manual* (TRM) for the *L220 MBIST Controller*.

Product revision status

The r*n*p*n* identifier indicates the revision status of the product described in this manual, where:

r*n* Identifies the major revision of the product.

p*n* Identifies the minor revision or modification status of the product.

Intended audience

This manual is written for hardware engineers who are familiar with ARM technology and want to use the L220 MBIST Controller to test the RAM blocks used by the L220 Cache Controller. The AXI protocol is not specified but some familiarity with AXI is assumed.

Using this manual

This manual is organized into the following chapters:

Chapter 1 *Introduction*

Read this chapter for an introduction to MBIST technology.

Chapter 2 *Functional Description*

Read this chapter for a description of the L220 interface to the L220 MBIST Controller and MBIST testing of the L220 data RAM, L220 data parity RAM, L220 tag RAMs, and L220 dirty RAM. Also read this chapter for a description of the timing sequences for loading MBIST instructions, starting the MBIST test, detecting failures, and retrieving the data log.

Chapter 3 *MBIST Instruction Register*

Read this chapter for a description on how to use the MBIST Instruction Register to configure the mode of operation of the MBIST engine.

Appendix A *Signal Descriptions*

Read this appendix for a description of the L220 MBIST Controller input and output signals.

Conventions

Conventions that this manual can use are described in:

- *Typographical*
- *Timing diagrams*
- *Signals* on page xiv
- *Numbering* on page xiv.

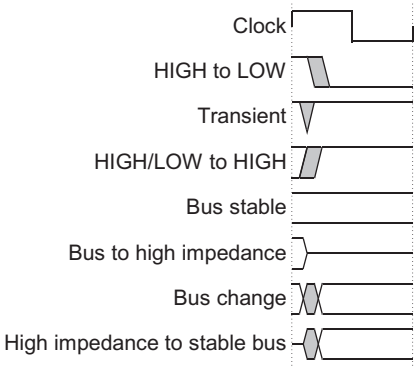
Typographical

The typographical conventions are:

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 you can enter at the keyboard, such as commands, file and program names, and source code.
`mono`space	Denotes a permitted abbreviation for a command or option. You can enter the underlined text instead of the full command or option name.
`monospace italic`	Denotes arguments to monospace text where the argument is to be replaced by a specific value.
`monospace bold`	Denotes language keywords when used outside example code.
< and >	Angle brackets enclose replaceable terms for assembler syntax where they appear in code or code fragments. They appear in normal font in running text. For example: • `MRC p15, 0 <Rd>, <CRn>, <CRm>, <Opcode_2>` • The Opcode_2 value selects the register that is accessed.

Timing diagrams

The figure named *Key to timing diagram conventions* on page xiv explains the components used in timing diagrams. Variations, when they occur, have clear labels. You must not assume any timing information that is not explicit in the diagrams.



Key to timing diagram conventions

Signals

The signal conventions are:

Signal level The level of an asserted signal depends on whether the signal is active-HIGH or active-LOW. Asserted means HIGH for active-HIGH signals and LOW for active-LOW signals.

Prefix A Denotes *Advanced eXtensible Interface* (AXI) global and address channel signals.

Prefix B Denotes AXI write response channel signals.

Prefix C Denotes AXI low-power interface signals.

Prefix n Denotes active-LOW signals.

Prefix R Denotes AXI read channel signals.

Prefix W Denotes AXI write channel signals.

Numbering

The numbering convention is:

<size in bits>'<base><number>

This is a Verilog method of abbreviating constant numbers. For example:

- 'h7B4 is an unsized hexadecimal value.
- 'o7654 is an unsized octal value.
- 8'd9 is an eight-bit wide decimal value of 9.

- 8'h3F is an eight-bit wide hexadecimal value of `0x3F`. This is equivalent to b00111111.
- 8'b1111 is an eight-bit wide binary value of b00001111.

Further reading

This section lists publications by ARM Limited, and by third parties.

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

ARM publications

This manual contains information that is specific to the L220 MBIST Controller. See the following documents for other relevant information:

- *AMBA AXI Protocol Specification* (ARM IHI 0022)
- *L220 Cache Controller Technical Reference Manual* (ARM DDI 0329)
- *L220 Cache Controller Implementation Guide* (ARM DII 0104).

Feedback

ARM Limited welcomes feedback on the L220 MBIST Controller and its documentation.

Feedback on this product

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 manual

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

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

ARM Limited also welcomes general suggestions for additions and improvements.

Chapter 1
Introduction

This chapter introduces the MBIST Controller. It contains the following sections:

1.1 About the L220 MBIST Controller

MBIST is the industry-standard method of testing embedded memories. MBIST works by performing sequences of reads and writes to the memory according to a test algorithm. Many industry-standard test algorithms exist.

An MBIST Controller generates the correct sequence of reads and writes to all locations of the RAM to ensure that the cells are operating correctly. In doing this, some additional test coverage is achieved in the address and data paths that the MBIST uses. You must only use the L220 MBIST Controller only with the L220 Cache Controller to perform memory testing of the <*Level 2*> cache RAM.

Note

The example integration files provided with the L220 MBIST Controller only support an 8-way cache design.

MBIST mode takes priority over all other modes for example SCAN, in that the L2 RAMs are only accessible to the L220 MBIST Controller when MBIST mode is activated with the **MTESTON** pin. You must keep the **MTESTON** signal LOW during functional mode.

The L220 MBIST Controller controls the MBIST testing of the L2 RAMs through the MBIST port of the L220 Cache Controller. Figure 1-1 shows the L220 Cache Controller MBIST configuration.

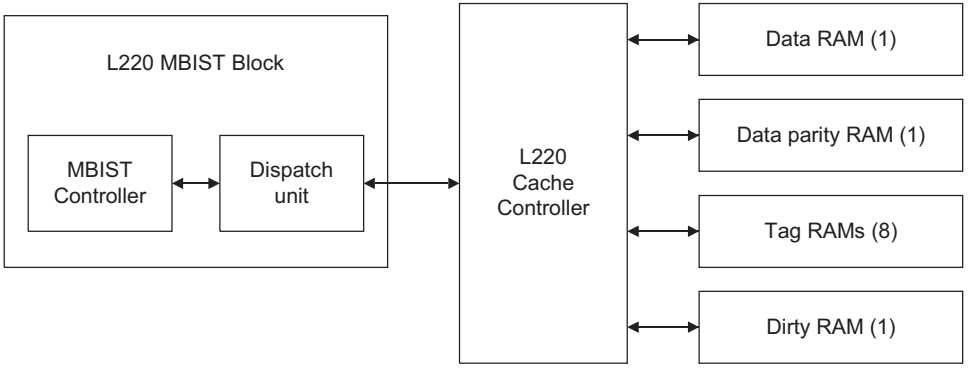


Figure 1-1 L220 Cache Controller MBIST configuration

1.2 L220 MBIST Controller interface

Figure 1-2 shows the L220 MBIST Controller interface to the *Automated Test Equipmen*t (ATE) and to the MBIST interface of the L220 Cache Controller.

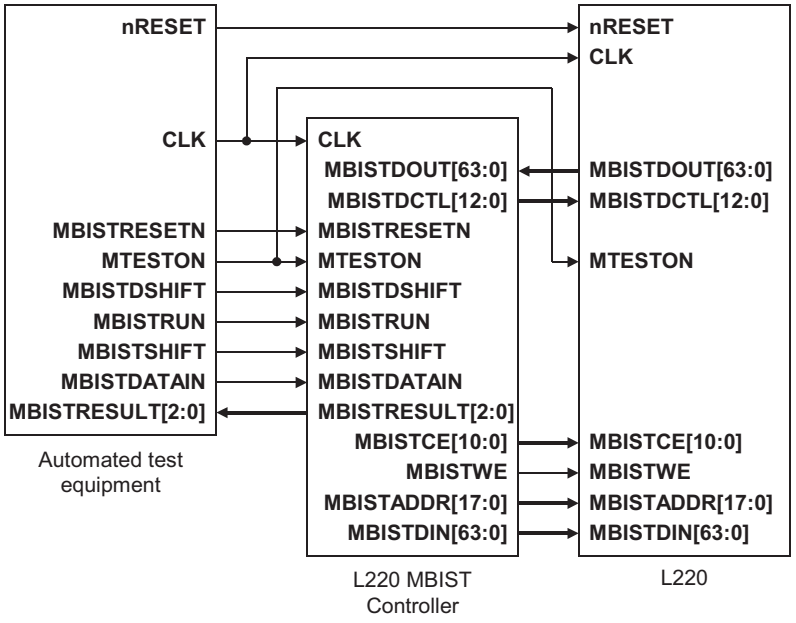


Figure 1-2 MBIST Controller wiring diagram

Figure 1-3 on page 1-4 shows the traditional method of accessing a cache RAM for MBIST.

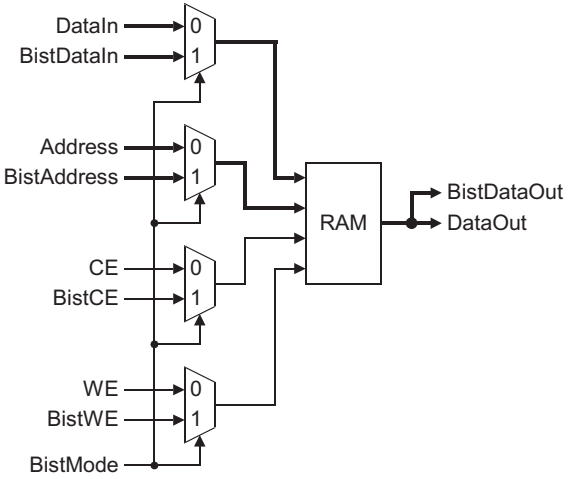


Figure 1-3 Traditional method of interfacing MBIST

Because this method significantly reduces the maximum operating frequency, it is not suitable for high-performance designs. Instead, the L220 MBIST Controller uses an additional input to the existing functional multiplexors without reducing maximum operating frequency.

Figure 1-4 on page 1-5 shows the five pipeline stages used to access the cache RAM arrays.

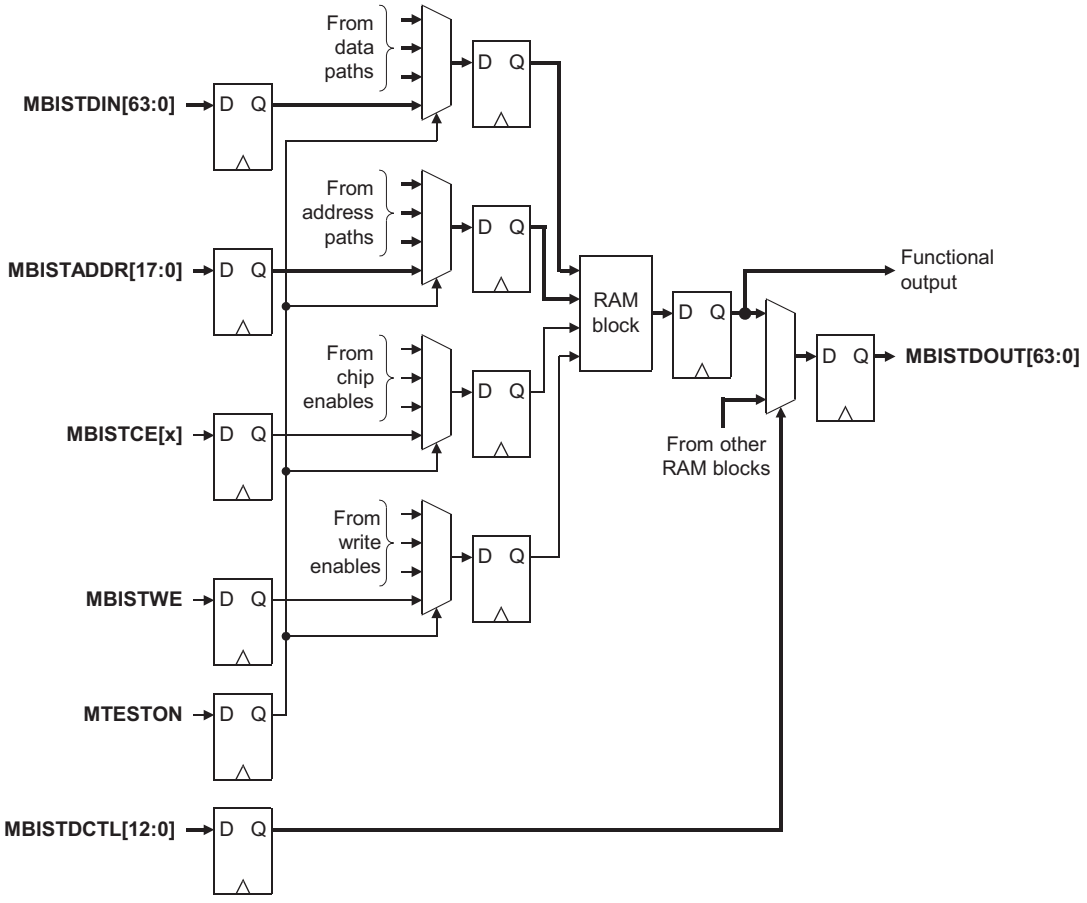


Figure 1-4 L220 Cache Controller MBIST interface

The L220 MBIST Controller accesses memory through the MBIST interface of the L220 Cache Controller. Table 1-1 lists the L220 Cache Controller MBIST interface signals.

Table 1-1 L220 Cache Controller MBIST interface signals

Name	Type	Description
nRESET	Input	Global active LOW reset signal.
CLK	Input	Active HIGH clock signal. This clock drives the L220 Cache Controller logic.
MBISTDOUT[63:0]	Output	Data out bus from all cache RAM blocks.

Table 1-1 L220 Cache Controller MBIST interface signals (continued)

Name	Type	Description
MBISTDCTL[12:0]	Input	Delayed versions of the **MBISTCE[10:0]** signal and the doubleword select signal, **MBISTADDR[1:0]**. Selects the correct read data after it passes through the MBIST pipeline stages. **MBISTDCTL[12:0]** = delayed {**MBISTCE[10:0]**, **MBISTADDR[1:0]**}.
MTESTON	Input	Select signal for cache RAM array. This signal is the select input to the multiplexors that access the cache RAM arrays for test. When asserted, **MTESTON** takes priority over all other select inputs to the multiplexors.
MBISTCE[10:0]	Input	One-hot chip enables to select cache RAM arrays for test.
MBISTWE	Input	Global write enable signal for all RAM arrays.
MBISTADDR[17:0]	Input	Address signal for cache RAM array. **MBISTADDR[1:0]** is the doubleword select value. See *Y-address and X-address fields, MBIR[23:20] and MBIR[27:24]* on page 3-8 for a description of the doubleword select. Not all RAM arrays use the full address width.
MBISTDIN[63:0]	Input	Data bus to the cache RAM arrays. Not all RAM arrays use the full data width.

Note

The interface of the L220 MBIST Controller communicates with both the ATE and the MBIST interface of the L220 Cache Controller. See Appendix A *Signal Descriptions* for descriptions of the L220 MBIST Controller interface signals. See the *L220 Cache Controller Technical Reference Manual* for more information about the MBIST interface.

1.3 Product revisions

This section describes differences in functionality between product revisions of the MBIST Controller:

r0p0-r1p2 Contains no differences in functionality. Revision status updated.

r1p2-r1p3 Contains no differences in functionality. Revision status updated.

r1p3-r1p4 Contains no differences in functionality. Revision status updated.

r1p4-r1p5 Contains no differences in functionality. Revision status updated.

r1p5-r1p7 Contains no differences in functionality. Revision status updated.

Chapter 2
Functional Description

This chapter contains a functional overview and L220 MBIST Controller implementation. The functional operation provides timing sequences for loading instructions, starting the MBIST engine, detecting failures, and retrieving the data log. It contains the following sections:

2.1 Functional overview

This section provides a block diagram of the L220 MBIST Controller, L220 Cache Controller, RAM interfaces and signal groups. It then provides a description of each block for testing with associated diagram and signal connections. Finally the section covers the L220 MBIST Controller implementation. This section describes:

- *MBIST Controller interface*
- *MBIST Controller implementation* on page 2-12.

2.1.1 MBIST Controller interface

The L220 MBIST Controller has one MBIST port, see Appendix A *Signal Descriptions*. Only one RAM is accessed by the L220 MBIST Controller at any time.

The MBIST Controller must be able to account for the different latencies of the RAMs. You can configure RAM latencies for the L220 Cache Controller RAMS. You can configure the following RAMs for up to eight cycles of latency:

- data read
- data write
- tag read and write
- dirty read and write
- parity read and write.

See also *L220 Compiled RAM Latencies* on page 2-3.

You can use the MBIST Controller for testing the L220 Cache Controller compiled RAMs. You can also choose to design your own MBIST Controller. You can only access one RAM by the MBIST port at a time.

The Tag Parity RAM, if present, is tested at the same time as the Tag RAMs. Parity bits are considered as an extra bit on the Tag Data Bus.

Figure 2-1 on page 2-3 shows the interfaces between the L220 MBIST Controller and the RAMs that MBIST tests.

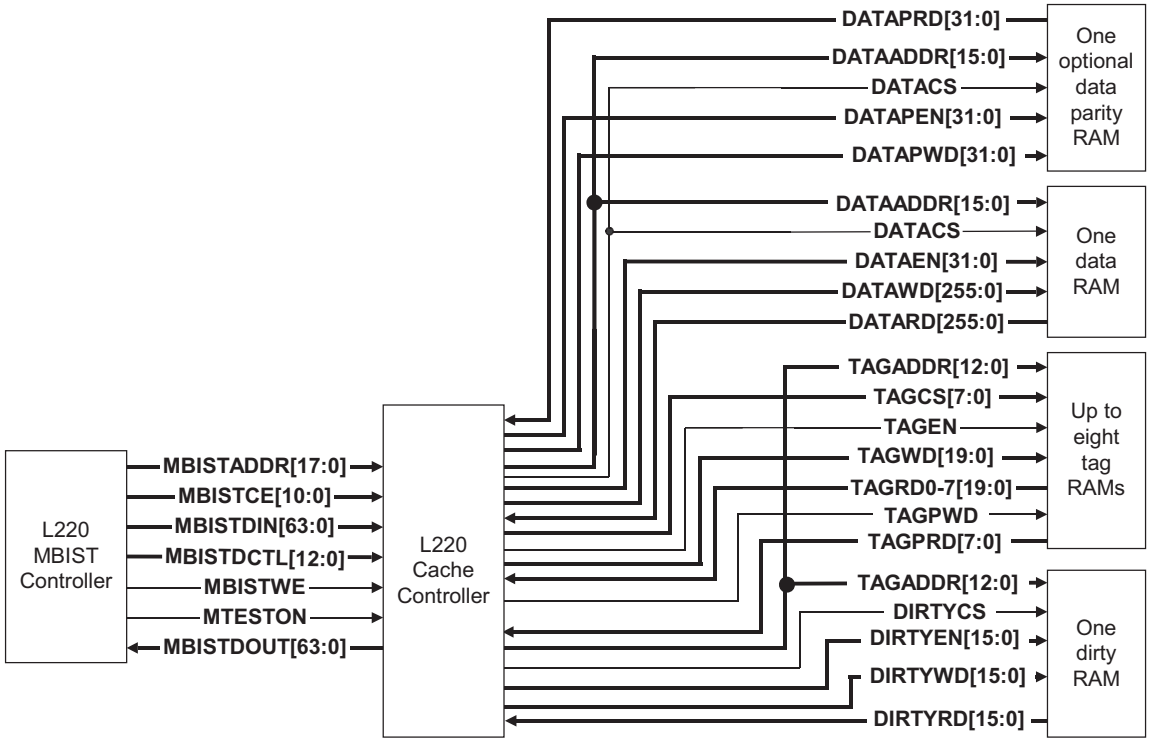


Figure 2-1 L220 Cache Controller MBIST and RAM Interfaces

This section describes RAM latencies and the four MBIST RAM tests:

- *L220 Compiled RAM Latencies*
- *MBIST testing of L220 Cache Controller data RAM* on page 2-4
- *MBIST Testing of L220 Cache Controller data parity RAM* on page 2-7
- *MBIST testing of L220 Cache Controller tag RAMs* on page 2-8
- *MBIST testing of L220 dirty RAM* on page 2-11.

L220 Compiled RAM Latencies

The L220 Cache Controller resets assuming the slowest compiled RAMs are being used. This means eight L220 Cache Controller clock cycles are used for each access. In terms of reads, this means that the read data is sampled eight clock edges after the edge on which the read request is sampled by the RAM. Using this nomenclature, the shortest latency is one. During functional mode, the latencies for each RAM are programmed in the L220 Cache Controller Auxiliary Control Register. For MBIST, you must know the latencies of the RAMs being tested. The MBIST Controller defaults to one cycle of

latency, but must reprogram this during the instruction load before MBIST testing can begin. The latency of the current RAM being tested is passed to the L220 MBIST Controller in the MBIST instruction. Table 2-1 shows the L220 Cache Controller compiled RAM latency.

Table 2-1 L220 Cache Controller compiled RAM Latency

Latency bits [2:0]	Cycles of latency
3'b000	1 cycle of latency. No additional latency. This is the default.
3'b001	2 cycles of latency.
3'b010	3 cycles of latency.
3'b011	4 cycles of latency.
3'b100	5 cycles of latency.
3'b101	6 cycles of latency.
3'b110	7 cycles of latency.
3'b111	8 cycles of latency.

Figure 2-2 shows the L220 Cache Controller compiled RAM latency.

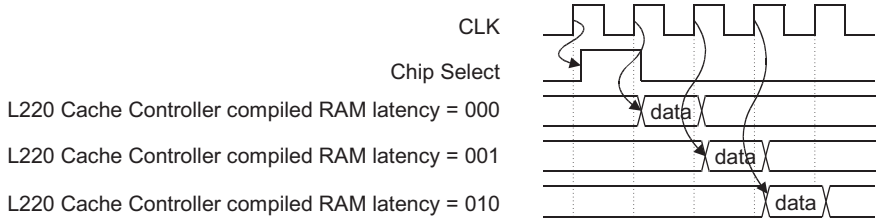


Figure 2-2 L220 Cache Controller compiled RAM Latency

MBIST testing of L220 Cache Controller data RAM

The L220 Cache Controller data RAM is 256 bits wide, and the size of the **MBISTDIN** and **MBISTDOUT** buses on the L220 Cache Controller MBIST interface is 64 bits, so four reads and four writes are required for each index of the data RAM. The L220 Cache Controller handles this by using the **MBISTADDR[1:0]** signal as a doubleword select for each index of the data RAM for writes. For reads from a previous MBIST transaction you use the **MBISTDCTL[1:0]** signal. You require separate pins because

the MBIST transactions are pipelined. The MBIST Controller takes into account the data RAM latency and issues the correct control signals. Table 2-2shows the address range of the **MBISTADDR** bus used to test the data RAM, based on the L2 cache size and configured to be 8-way.

Table 2-2 MBISTADDR and MBISTDIN mapping for data RAM, 8-way

L2 cache size	Number of data RAM indexes	MBISTADDR to data RAM mapping	MBISTDIN to data RAM mapping
128KB	4,096	**DATAADDR[11:0]=MBISTADDR[17:15,10:2]**	**DATAWD[63:0]=MBISTDIN[63:0]**
256KB	8,192	**DATAADDR[12:0]=MBISTADDR[17:15,11:2]**	**DATAWD[63:0]=MBISTDIN[63:0]**
512KB	16,384	**DATAADDR[13:0]=MBISTADDR[17:15,12:2]**	**DATAWD[63:0]=MBISTDIN[63:0]**
1MB	32,768	**DATAADDR[14:0]=MBISTADDR[17:15,13:2]**	**DATAWD[63:0]=MBISTDIN[63:0]**
2MB	65,536	**DATAADDR[15:0]=MBISTADDR[17:2]**	**DATAWD[63:0]=MBISTDIN[63:0]**

For a 4-way cache, you can remove one bit from the upper address range and add it to the lower address range as compared to an 8-way cache of the same size. Table 2-3shows the address range of the **MBISTADDR** bus used to test the data RAM, based on the L2 cache size and configured to be 4-way.

Table 2-3 MBISTADDR and MBISTDIN mapping for data RAM, 4-way

L2 cache size	Number of data RAM indexes	MBISTADDR to data RAM mapping	MBISTDIN to data RAM mapping
128KB	4,096	**DATAADDR[11:0]=MBISTADDR[16:15,11:2]**	**DATAWD[63:0]=MBISTDIN[63:0]**
256KB	8,192	**DATAADDR[12:0] =MBISTADDR[16:15,12:2]**	**DATAWD[63:0] = MBISTDIN[63:0]**
512KB	16,384	**DATAADDR[13:0] =MBISTADDR[16:15,13:2]**	**DATAWD[63:0] = MBISTDIN[63:0]**
1MB	32,768	**DATAADDR[14:0]=MBISTADDR[16:2]**	**DATAWD[63:0] = MBISTDIN[63:0]**

Note

A 2M 4-way cache is not possible because the L220 Cache Controller maximum way size is limited to 256KB.

The L220 Cache Controller has a *Line Read Buffer* (LRB), in each slave these are 256 bits wide. One of these holds data for MBIST testing. The L220 Cache Controller always adds two register delays to the MBIST data read path for the data RAM.

When using the L220 MBIST Controller you must account for the data RAM latency in the pipeline. The latency can be from one to eight clock cycles. See *L220 Compiled RAM Latencies* on page 2-3. The signal **MBISTCE[0]** is for the chip enable to the data RAM. The signal **MBISTDCTL[2:0]** is for reads from previous MBIST transactions.

Figure 2-3 shows the L220 Cache Controller MBIST paths for data RAM testing.

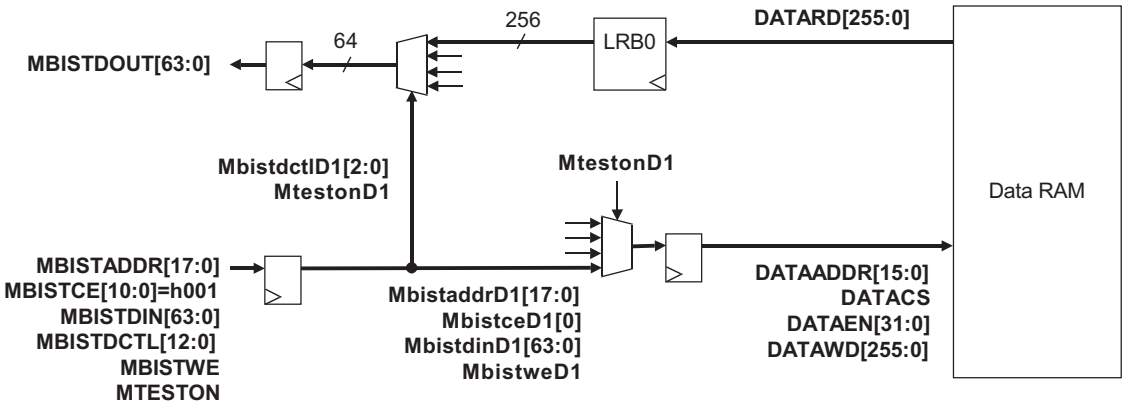


Figure 2-3 L220 Cache Controller MBIST paths for data RAM testing

Table 2-4shows the write sequences for data RAM testing.

Table 2-4 Writes for data RAM testing

MBISTADDR[1:0]	DATAEN[31:0]	DATAWD used
b00	0x000F	[63:0]
b01	0x00F0	[127:64]
b10	0x0F00	[191:128]
b11	0xF000	[255:192]

MBIST Testing of L220 Cache Controller data parity RAM

There is one data parity RAM associated with the L220. Table 2-5 shows the address range of the **MBISTADDR** bus used to test the data parity RAM, based on the L2 cache size and configured to be 8-way.

Table 2-5 MBISTADDR and MBISTDIN mapping for data parity RAM, 8-way

L2 cache size	Number of data RAM indexes	MBISTADDR to data parity RAM mapping	MBISTDIN to data parity RAM mapping
128KB	4,096	**DATAADDR[11:0]=MBISTADDR[17:15,10:2]**	**DATAPWD[31:0]=MBISTDIN[31:0]**
256KB	8,192	**DATAADDR[12:0]=MBISTADDR[17:15,11:2]**	**DATAPWD[31:0]=MBISTDIN[31:0]**
512KB	16,384	**DATAADDR[13:0]=MBISTADDR[17:15,12:2]**	**DATAPWD[31:0]=MBISTDIN[31:0]**
1MB	32,768	**DATAADDR[14:0]=MBISTADDR[17:15,13:2]**	**DATAPWD[31:0]=MBISTDIN[31:0]**
2MB	65,536	**DATAADDR[15:0]=MBISTADDR[17:2]**	**DATAPWD[31:0]=MBISTDIN[31:0]**

The L220 Cache Controller has an LRB, in each slave these are each 256 bits wide. One of these holds data for MBIST testing. The L220 Cache Controller always adds two register delays to the MBIST data read path for the data parity RAM.

When using the L220 MBIST Controller you must account for the data parity RAM latency in the pipeline. The latency can be from one to eight clock cycles. See *L220 Compiled RAM Latencies* on page 2-3. The signal **MBISTCE[10]** is for the chip enable to the data parity RAM. The signal **MBISTDCTL[12]** is for reads from previous MBIST transactions.

Figure 2-4 on page 2-8 shows the L220 Cache Controller MBIST paths for data parity RAM testing.

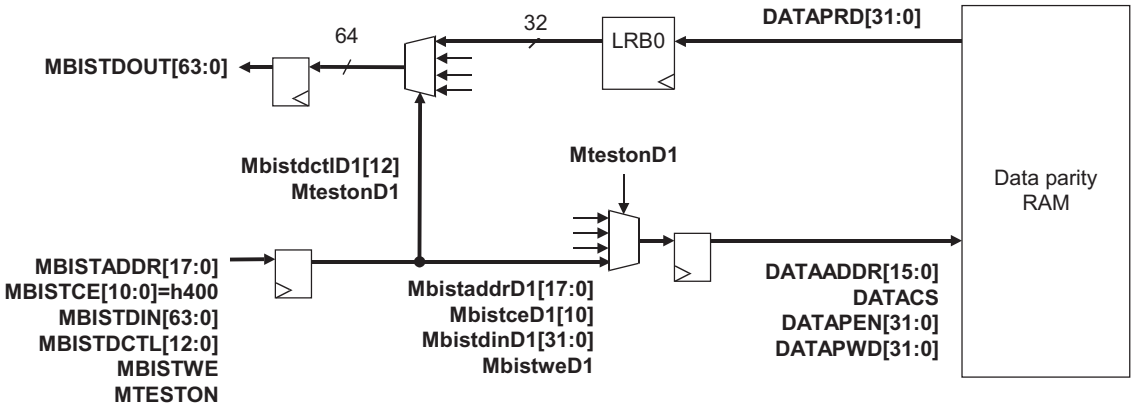


Figure 2-4 L220 Cache Controller MBIST paths for data parity RAM testing

MBIST testing of L220 Cache Controller tag RAMs

There is one Tag RAM for each way of the L2 cache. The maximum number of tag RAMs the MBIST Controller has to test is eight. Only one tag RAM is tested at a time. Table 2-6 shows the address range of the **MBISTADDR** bus used to test a tag RAM, based on the L2 cache size and configured to be 8-way. The parity for each tag RAM present is tested along with the rest of the tag and is mapped to **MBISTDIN[20]**.

Table 2-6 MBISTADDR and MBISTDIN mapping for tag RAM, 8-way

L2 cache size	Number of tag RAM indexes	MBISTADDR to tag RAM mapping	MBISTDIN to tag RAM mapping
128KB	512	**TAGADDR[8:0]=MBISTADDR[10:2]**	**TAGWD[20:0]=MBISTDIN[19:0,20]**
256KB	1,024	**TAGADDR[9:0]=MBISTADDR[11:2]**	**TAGWD[19:0]=MBISTDIN[19:1,20]**
512KB	2,048	**TAGADDR[10:0]=MBISTADDR[12:2]**	**TAGWD[18:0]=MBISTDIN[19:2,20]**
1MB	4,096	**TAGADDR[11:0]=MBISTADDR[13:2]**	**TAGWD[17:0]=MBISTDIN[19:3,20]**
2MB	8,192	**TAGADDR[12:0]=MBISTADDR[14:2]**	**TAGWD[16:0]=MBISTDIN[19:4,20]**

Table 2-7 shows the address range of the **MBISTADDR** bus used to test the tag RAM, based on the L2 cache size and configured to be 4-way.

Table 2-7 MBISTADDR and MBISTDIN mapping for tag RAM, 4-way

L2 cache size	Number of tag RAM indexes	MBISTADDR to tag RAM mapping	MBISTDIN to tag RAM mapping
128KB	4,096	**TAGADDR[9:0]=MBISTADDR[11:2]**	**TAGWD[19:0]=MBISTDIN[19:1,20]**
256KB	8,192	**TAGADDR[10:0]=MBISTADDR[12:2]**	**TAGWD[18:0]=MBISTDIN[19:2,20]**
512KB	16,384	**TAGADDR[11:0]=MBISTADDR[13:2]**	**TAGWD[17:0]=MBISTDIN[19:3,20]**
1MB	2,768	**TAGADDR[12:0]=MBISTADDR[14:2]**	**TAGWD[16:0]=MBISTDIN[19:4,20]**

The data from the tag RAMs is always registered by the L220 Cache Controller, plus there is a register on the MBIST port of the L220 Cache Controller. Consequently, to the L220 MBIST Controller, the L220 always adds two register delays to the MBIST data read path for the tag RAMs.

When using the L220 MBIST Controller you must account for the Tag RAM latency in the pipeline. The signal **MBISTCE[8:1]** is for chip enables to the tag RAMs. The signal **MBISTDCTL[10:3]** is for reads from previous MBIST transactions. The latency of the tag RAMs can be from one to eight clock cycles. See *L220 Compiled RAM Latencies* on page 2-3.

Figure 2-5 on page 2-10 shows the L220 Cache Controller MBIST paths for tag RAM testing.

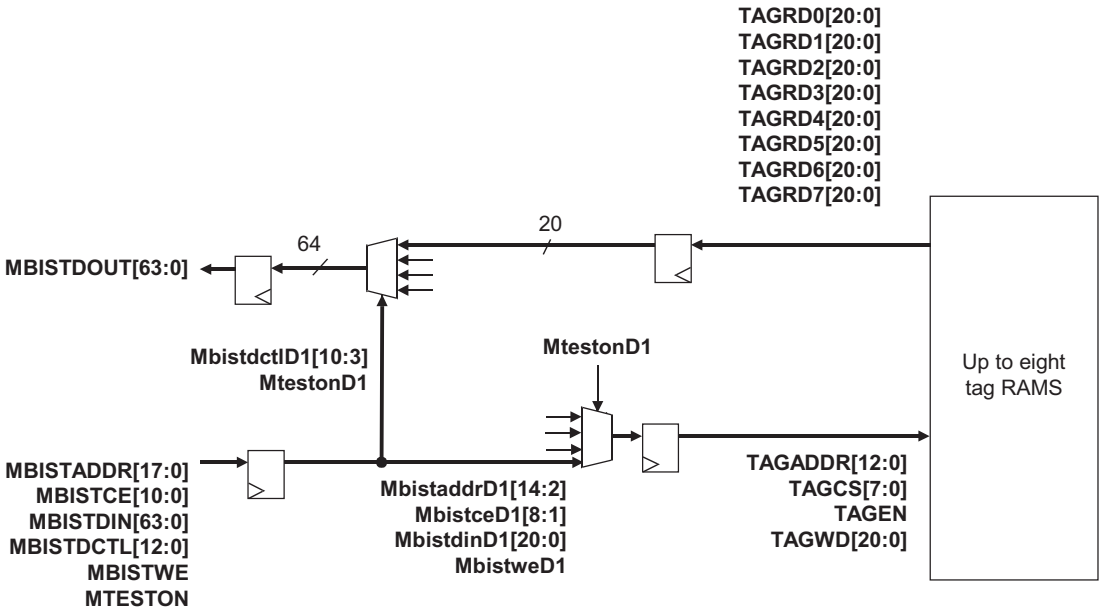


Figure 2-5 L220 Cache Controller MBIST paths for tag RAM testing

Note

- **MBISTCE[8:1]** corresponds to **TAGCS[7:0]**
- **MDBISTDCL[10:3**] corresponds to **TAG[7:0]**
- Only [20:0] of **MBISTDIN** and **MBISTDOUT** are used.

MBIST testing of L220 dirty RAM

There is one dirty RAM associated with the L220 Cache Controller. The dirty RAM uses the same address as the tag RAMs. Table 2-8 shows the address range of the **MBISTADDR** bus used to test the dirty RAM, based on the L2 cache size and configured to be 8-way.

Table 2-8 MBISTADDR and MBISTDIN mapping for dirty RAM, 8-way

L2 cache size	Number of dirty RAM indexes	MBISTADDR to dirty RAM mapping	MBISTDIN to dirty RAM mapping
128KB	512	**TAGADDR[8:0]=MBISTADDR[10:2]**	**DIRTYWD[15:0]=MBISTDIN[15:0]**
256KB	1,024	**TAGADDR[9:0]=MBISTADDR[11:2]**	**DIRTYWD[15:0]=MBISTDIN[15:0]**
512KB	2,048	**TAGADDR[10:0]=MBISTADDR[12:2]**	**DIRTYWD[15:0]=MBISTDIN[15:0]**
1MB	4,096	**TAGADDR[11:0]=MBISTADDR[13:2]**	**DIRTYWD[15:0]=MBISTDIN[15:0]**
2MB	8,192	**TAGADDR[12:0]=MBISTADDR[14:2]**	**DIRTYWD[15:0]=MBISTDIN[15:0]**

Table 2-9 shows the address range of the **MBISTADDR** bus used to test the dirty RAM, based on the L2 cache size and configured to be 4-way.

Table 2-9 MBISTADDR and MBISTDIN mapping for dirty RAM, 4-way

L2 cache size	Number of Dirty RAM Indexes	MBISTADDR to dirty RAM Mapping	MBISTDIN to dirty RAM Mapping
128KB	4,096	**TAGADDR[9:0]=MBISTADDR[11:2]**	**DIRTYWD[15:0]=MBISTDIN[15:0]**
256KB	8,192	**TAGADDR[10:0]=MBISTADDR[12:2]**	**DIRTYWD[15:0]=MBISTDIN[15:0]**
512KB	16,384	**TAGADDR[11:0]=MBISTADDR[13:2]**	**DIRTYWD[15:0]=MBISTDIN[15:0]**
1MB	2,768	**TAGADDR[12:0]=MBISTADDR[14:2]**	**DIRTYWD[15:0]=MBISTDIN[15:0]**

The data from the dirty RAM is always registered by the L220 Cache Controller, plus there is a register on the MBIST port of the L220 Cache Controller. Consequently, to the L220 MBIST Controller, the L220 always adds two register delays to the MBIST data read path for the dirty RAM.

When using the L220 MBIST Controller you must account for the dirty RAM latency in the pipeline. The signal **MBISTCE[9]** is for the chip enable to the dirty RAM. The signal **MBISTDCTL[11]** is for reads from previous MBIST transactions. The latency can be from one to eight clock cycles. See *L220 Compiled RAM Latencies* on page 2-3.

Figure 2-6 shows the L220 Cache Controller MBIST paths for dirty RAM testing.

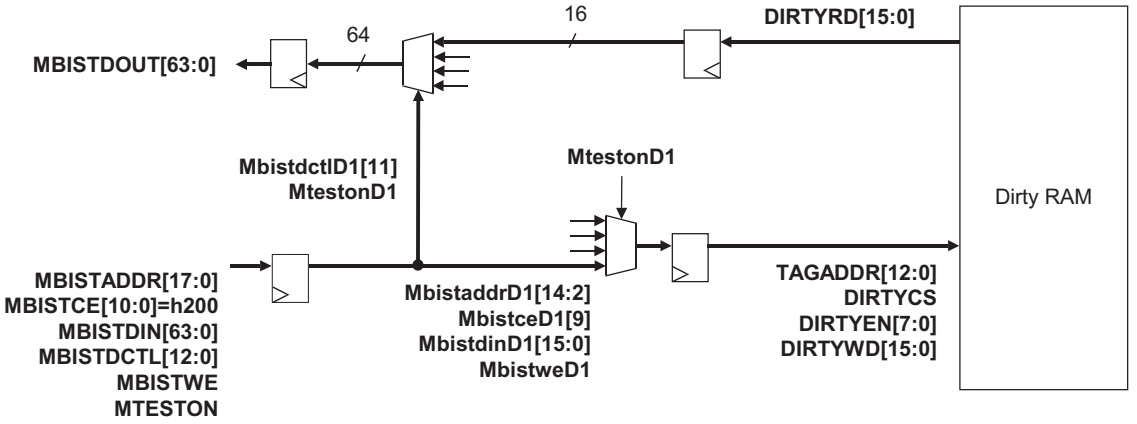


Figure 2-6 L220 Cache Controller MBIST paths for dirty RAM testing

Note

- **MBISTCE[9]** corresponds to **DIRTYCS**
- **MBISTDCTL[11]** corresponds to the dirty RAM
- Only [15:0] of **MBISTDIN** and **MBISTOUT** are used
- The dirty RAM uses the same address as the tag RAM.

2.1.2 MBIST Controller implementation

The L220 MBIST Controller block that Figure 2-7 on page 2-13 shows, consists of two major blocks:

- MBIST Controller
- Dispatch unit.

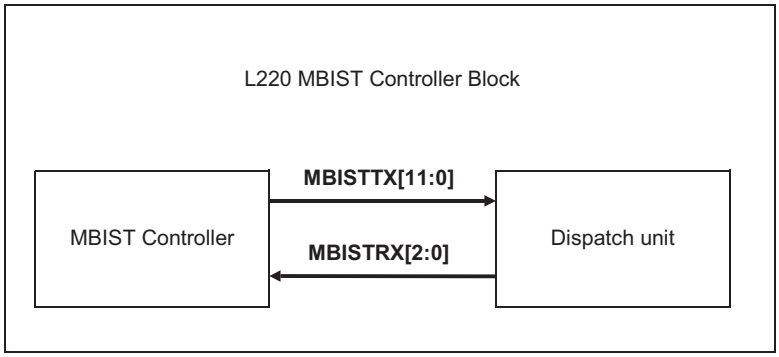


Figure 2-7 L220 MBIST Controller block

This section is subdivided into:

- *MBIST Controller and dispatch unit interface*
- *L220 MBIST Controller block top level I/O* on page 2-15.

MBIST Controller and dispatch unit interface

The MBIST Controller and the dispatch unit communicate using the following signals:

MBISTTX[11:0]

This signal is an output of the MBIST Controller that goes to the dispatch unit. Table 2-10 shows the signals.

Table 2-10 MBISTTX signals

MBISTTX bit	Description
0	Reset address
1	Increment address
2	Access sacrificial row, used during bang patterns
3	Invert data/instruction data in
4	Checkerboard data
5	Write data
6	Read data
7	Yfast/nXfast

Table 2-10 MBISTTX signals (continued)

MBISTTX bit	Description
8	Direction
9	Enable bitmap mode
10	Increment go/nogo dataword selection
11	Latency stall control

When the instruction shift is enabled, data shifts in on bit 3, normally Invert data and shifts into the instruction scan chain of the dispatch unit. The **MBISTTX[11:0]** interface is ARM-specific and intended for use only with the L220 MBIST Controller.

MBISTRX[2:0]

This signal is an output of the dispatch unit that goes to the MBIST Controller. The behavior of **MBISTRX[2:0]** is ARM-specific and is intended for use only with the L220 MBIST Controller. The address expire signal is set when both the row and column address counters expire. Table 2-11 shows the signals.

Table 2-11 MBISTRX signals

MBISTRX bit	Description
0	Address/instruction data out/fail data out
1	Shadow pipeline empty
2	Nonsticky fail flag

L220 MBIST Controller block top level I/O

The top level I/O of the L220 MBIST Controller consists of the L220 interface. See Appendix A *Signal Descriptions* and the inputs and outputs that Table 2-12 shows.

Table 2-12 L220 MBIST Controller top level I/O

Signal	Direction	Function	Value, MBIST mode	Value, function mode
MBISTDATAIN	Input	Serial data in	Toggle	0
MBISTDSHIFT	Input	Data log shift	Toggle	0
MBISTRESETN	Input	MBIST reset	Toggle	0[a]
MBISTRESULT[2:0]	Output	Output status bus	Strobe	-
MBISTRUN	Input	Run MBIST test	Toggle	0
MBISTSHIFT	Input	Instruction shift	Toggle	0
MTESTON	Input	MBIST path enable	Toggle	0
SCANENABLE	Input	ATPG signal	0	0
SCANMODE	Input	ATPG signal	0	0

a. **MBISTRESETN** and **MTESTON** must be LOW in functional mode

Note

nRESET of the L220 Cache Controller must be HIGH in MBIST test mode.

The following signals have additional information:

SCANENABLE

Preservation of array state is required when performing multiload ATPG runs or when performing **IDDQ** testing. The L220 MBIST Controller keeps all L220 RAM chip select signals LOW with the **SCANENABLE** signal. After performing MBIST tests to initialize the arrays to a required background, the *Automatic Test Pattern Generator* (ATPG) test procedures must assert **SCANEANABLE** during all test setup cycles in addition to load/unload. Any clocking during **IDDQ** capture cycles must have array chip select signals constrained.

MBISTRESULT[2:0]

During tests, the **MBISTRESULT[1]** signal indicates failures. You can operate using two modes, by configuring bit 5 of the engine control section of the instruction register. If bit 5 is set, the **MBISTRESULT[1]** signal is asserted for a single cycle for each failed compare. If bit 5 is not set, the **MBISTRESULT[1]** signal is sticky, and is asserted from the first failure until the end of the test.

At the completion of the test, the **MBISTRESULT[2]** signal goes HIGH. The **MBISTRESULT[0]** signal indicates that an address expire has occurred and enables you to measure sequential progress through the test algorithms.

2.2 Functional operation

The functional operation is described in:

- *Timing*
- *Bitmap mode* on page 2-20.

2.2.1 Timing

A 46-bit instruction, loaded serially at the start of each test, controls the operation of the L220 MBIST Controller. Chapter 3 *MBIST Instruction Register* describes how to write the instruction.

The timing diagrams in this section show the clock running at two different speeds:

- the slower clock relates to the clock driven by your ATE
- the faster clock relates to the clock driven by an on-chip *Phase Locked Loop* (PLL).

If you do not have an on-chip PLL, both clocks relate to the clock driven by your ATE.

Timing diagrams in the following sections show the procedures for operating the L220 MBIST Controller:

- *Instruction load*
- *Starting MBIST* on page 2-18
- *Failure detection* on page 2-18
- *Data log retrieval* on page 2-18.

Instruction load

To load an MBIST instruction, drive **MBISTSHIFT** HIGH. At the next rising clock edge, the 46-bit shift sequence begins as shown in Figure 2-8. To enable data input from the ATE, the PLL is in bypass mode, and the clock is not running at test frequency.

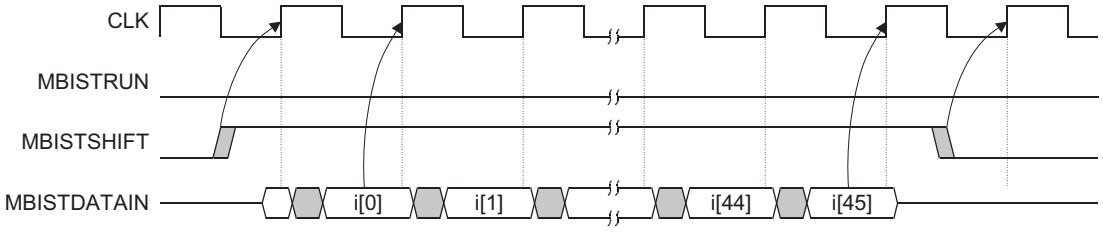


Figure 2-8 Loading the L220 MBIST Controller instruction

Starting MBIST

After loading the MBIST instruction, drive **MBISTSHIFT** LOW and disable **CLK**. With **CLK** disabled, drive **MBISTRUN** HIGH and, after an **MBISTRUN** setup time, start the PLL at the test frequency as shown in Figure 2-9.

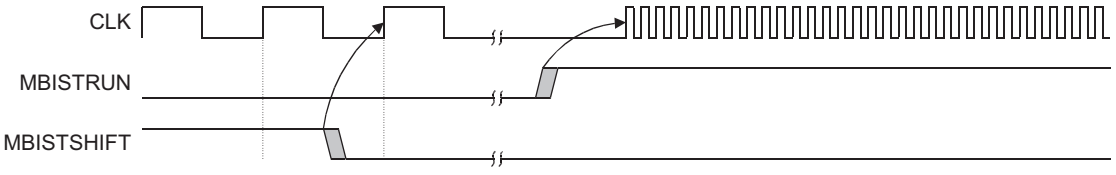


Figure 2-9 Starting the MBIST test

Failure detection

The **MBISTRESULT[1]** flag goes HIGH two **CLK** cycles after the controller detects a failure, as Figure 2-10 shows. It stays HIGH if sticky fail is enabled. If stop on fail is enabled, the **MBISTRESULT[2]** flag goes HIGH two cycles later.

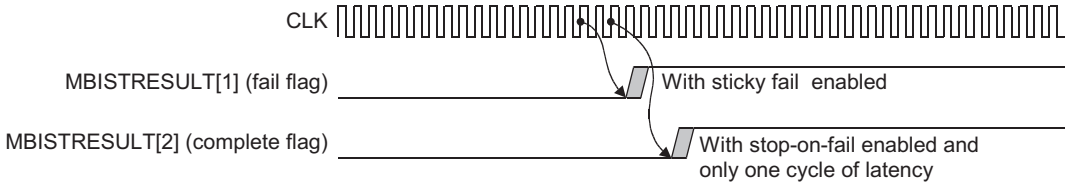


Figure 2-10 Detecting an MBIST failure

Note

To ensure that the ATE can observe a failure at test speed, specify a sticky fail in the MBIST instruction. See *Control field, MBIR[39:34]* on page 3-6.

Data log retrieval

During a test, the L220 MBIST Controller automatically logs the first detected failure. If required, you can retrieve the data log at the end of the test to generate failure statistics. Figure 2-11 on page 2-19 and Figure 2-12 on page 2-19 show the method of retrieving a data log.

Note

MBISTRESULT[0] is the serial data output for instructions and the data log.

After the **MBISTRESULT[2**] flag goes HIGH, stop the test by putting the PLL in bypass mode and driving **MBISTRUN** LOW as Figure 2-11 shows. To begin shifting out the data log on **MBISTRESULT[0]**, drive **MBISTDSHIFT** HIGH. The **MBISTRESULT[2]** flag goes LOW two cycles after **MBISTRUN** goes LOW. Data begins shifting out on **MBISTRESULT[0]** two cycles after **MBISTDSHIFT** goes HIGH.

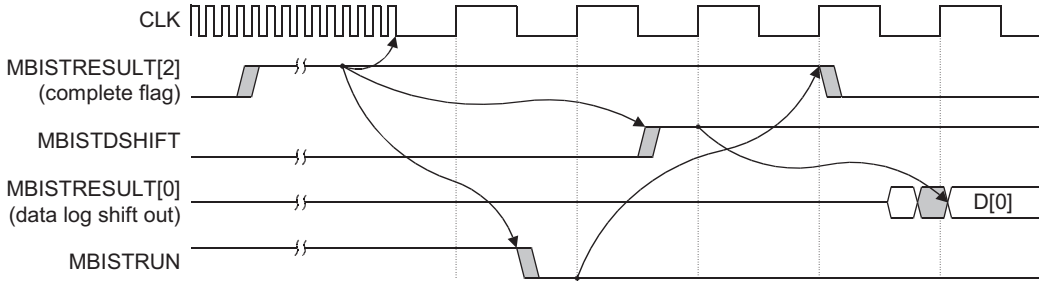


Figure 2-11 Start of data log retrieval

When the last data log bit shifts out, drive **MBISTDSHIFT** LOW as Figure 2-12 shows.

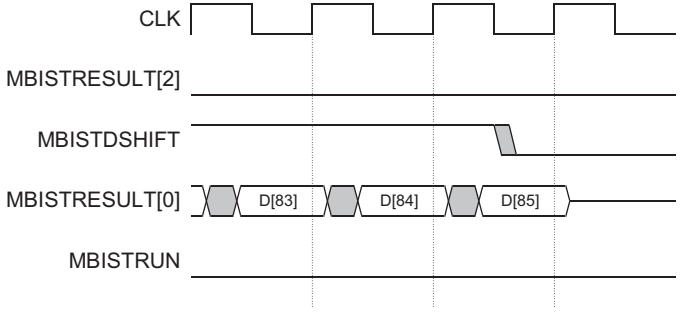


Figure 2-12 End of data log retrieval

Table 2-13 shows the format of the data log.

Table 2-13 Data log format

Bits	Description
[85:68]	Address of the failing location.
[67:4]	Failing data bits. These bits are set for faulty bits and cleared for passing bits.
[3:0]	The data seed used in the test. See *Data seed field, MBIR[19:16]* on page 3-12.

The address contained in the data log refers to the full address of the failing location as it appears on the **MBISTADDR[17:0]** port of the MBIST interface of the L220. It always includes the doubleword select value in the least significant two bits. See *Y-address and X-address fields, MBIR[23:20] and MBIR[27:24]* on page 3-8 for more information on the doubleword select value. Contact ARM Limited if you require more information.

2.2.2 Bitmap mode

In bitmap mode, you can identify all failing locations in a RAM. Each time a failure occurs, the controller stops executing the current test and waits for you to begin shifting out the data log as Figure 2-13 shows.

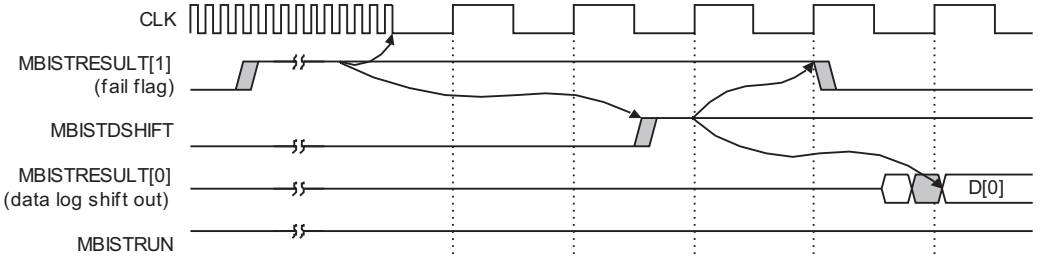


Figure 2-13 Start of bitmap data log retrieval

After you finish shifting and drive **MBISTDSHIFT** LOW, the controller then resumes testing where it stopped as Figure 2-14 on page 2-21 shows. This process continues until the test algorithm completes. A fault can cause a failure to occur several times during a given test algorithm. The fault might be logged multiple times depending on the number of reads performed by the algorithm and the exact nature of the fault.

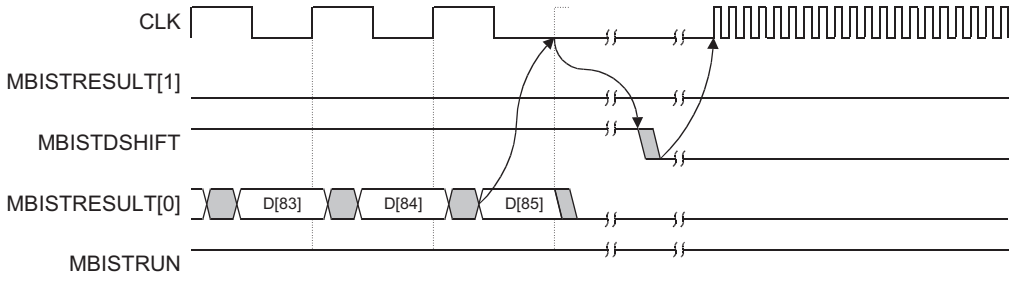


Figure 2-14 End of bitmap data log retrieval

Loading a new instruction resets bitmap mode.

Chapter 3
MBIST Instruction Register

This chapter describes how to use the *MBIST Instruction Register* (MBIR) to configure the mode of operation of the L220 MBIST Controller. It contains the following sections:

3.1 About the MBIST Instruction Register

Figure 3-1 shows the bit fields of the MBIR.

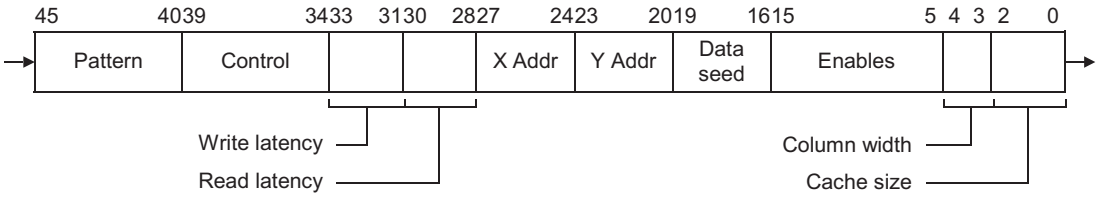


Figure 3-1 MBIST Instruction Register

The MBIR fields set up the behavior of the MBIST engine:

Pattern Specifies the test algorithm.

Control Specifies MBIST mode of operation and sticky or nonsticky fail flag.

Write latency Specifies the number of cycles to enable a RAM write.

Read latency Specifies the number of cycles to enable a RAM read.

X addr Specifies the number of bits in the X-address counter.

Y addr Specifies the number of bits in the Y-address counter.

Data seed Specifies the four-bit data background.

Enables Specifies the RAM under test.

Column width Specifies 4, 8, 16, or 32 columns per block of RAM.

Cache size Specifies a cache size of 128KB, 256KB, 512KB, 1MB, or 2MB.

Field descriptions on page 3-3 describes the MBIR fields in more detail.

3.2 Field descriptions

The following sections describe the MBIR fields:

- *Pattern field, MBIR[45:40]*
- *Control field, MBIR[39:34]* on page 3-6
- *Read latency and write latency fields, MBIR[30:28] and MBIR[33:31]* on page 3-6
- *Y-address and X-address fields, MBIR[23:20] and MBIR[27:24]* on page 3-8
- *Data seed field, MBIR[19:16]* on page 3-12
- *Enables field, MBIR[15:5]* on page 3-13
- *Column width field, MBIR[4:3]* on page 3-13
- *Cache size field, MBIR[2:0]* on page 3-14.

3.2.1 Pattern field, MBIR[45:40]

The L220 MBIST Controller is supplied with industry-standard pattern algorithms and a bit-line stress algorithm. You can group algorithms together to create a specific memory test methodology for your product.

Table 3-1 describes the supported algorithms, and *Pattern specification* on page 3-4 describes their use. The N values in the table indicate the number of RAM accesses per address location and give an indication of the test time when using that algorithm.

Table 3-1 Pattern field encoding

Pattern MBIR[45:40]	Algorithm name	N	Description
b000000	Write Solids	1N	Write a solid pattern to memory
b000001	Read Solids	1N	Read a solid pattern from memory
b000010	Write Checkerboard	1N	Write a checkerboard pattern to memory
b000011	Read Checkerboard	1N	Read a checkerboard pattern from memory
b000100	March C+ (x-fast)	14N	March C+ algorithm, incrementing X-address first
b001011	March C+ (y-fast)	14N	March C+ algorithm, incrementing Y-address first
b000101	Fail Pattern	6N	Tests memory failure detection capability
b000110	Read Write March (x-fast)	6N	Read write march pattern, incrementing X-address first
b000111	Read Write March (y-fast)	6N	Read write march pattern incrementing Y-address first

Table 3-1 Pattern field encoding (continued)

Pattern MBIR[45:40]	Algorithm name	N	Description
b001000	Read Write Read March (x-fast)	8N	Read write read march pattern, incrementing X-address first
b001001	Read Write Read March (y-fast)	8N	Read write read march pattern, incrementing y-address first
b001010	Bang	18N	Bit-line stress pattern
b111111	Go/No-Go	30N	See Table 3-2 on page 3-5

Pattern specification

This section describes the MBIST test patterns. An x-fast pattern increments or decrements the X-address counter first. A y-fast pattern increments or decrements the Y-address counter first. *Y-address and X-address fields, MBIR[23:20] and MBIR[27:24]* on page 3-8 describes the X-address and Y-address counters.

The first four patterns are useful for data retention or I_{DDQ} testing.

Write Solids This initializes the RAM with the supplied data seed.

Read Solids This reads each RAM location once expecting the supplied data seed.

Write Checkerboard

This initializes the RAM with a physical checkerboard pattern created by alternating the supplied data seed and its inverse.

Read Checkerboard

This reads back the physical checkerboard pattern created by alternating the supplied data seed and its inverse.

For the next set of patterns, the following notation describes the algorithm:

- 0 represents the data seed
- 1 represents the inverse data seed
- w indicates a write operation
- r indicates a read operation
- ⇑indicates that the address is incremented
- ⇓ indicates that the address is decremented.

March C+ (x-fast or y-fast)

This is the industry-standard March C+ algorithm:

`(w0) (r0, w1, r1) (r1, w0, r0) ⇓ (r0, w1, r1) ⇓ (r1, w0, r0) (r0)`

Read Write March (x-fast or y-fast)

`(w0) (r0, w1) ⇓ (r1, w0) (r0)`

Read Write Read March (x-fast or y-fast)

`(w0) (r0, w1, r1) ⇓ (r1, w0, r0) (r0)`

Bang This test is always performed in x-fast. It executes multiple consecutive writes and reads effectively stressing a bit-line pair. While this pattern does detect stuck-at faults, its primary intent is to address the analog characteristics of the memory. In the following algorithm description, row 0 indicates a read or write of the data seed to the sacrificial row, this is always the first row of the column being addressed.

`(w0) (r0, w0, w0(row 0) × 6)  (r0 × 5, w0(row 0), r0) (r0)`

Go/No-Go If you do not want to implement your own memory test strategy, use the Go/No-Go test pattern that performs the algorithms that Table 3-2 shows.

Table 3-2 Go/No-Go test pattern

Sequence	Algorithm	Data
1	Write Checkerboard	Data seed
2	Read Checkerboard	Data seed
3	Write Checkerboard	Data seed
4	Read Checkerboard	Data seed
5	Read Write Read March (y-fast)	`0x6`
6	Bang	`0xF`

This test suite provides a comprehensive test of the arrays. The series of tests in Go/No-Go are the result of the experience in memory testing by ARM memory test engineers.

3.2.2 Control field, MBIR[39:34]

The control field specifies the MBIST function. Table 3-3 shows how the control field affects the behavior of the L220 MBIST Controller.

Table 3-3 Control field encoding

Control MBIR[39:34]	Behavior	Description
bx00000	Default	Test runs to completion. If MBIR[39] = 0, sticky fail present after first failure.
bx00001	Stop on fail	End of test on failure.
bx00011	Bitmap mode	Enables logging of each failure. See *Bitmap mode* on page 2-20.

MBIR[39] selects a nonsticky or sticky fail flag, **MBISTRESULT[1]**:

- When **MBIR[39**] is set, the fail bit toggles in real time. It goes HIGH for failing comparisons and LOW for passing comparisons.

  Note

  Setting **MBIR[39]** can cause the fail bit to toggle at the test frequency. It is not recommended when the external pin or the ATE cannot follow the test frequency.

- When **MBIR[39]** is cleared, the fail bit is sticky. It remains HIGH after the first failure until a new MBIST instruction shifts in or until the data log shifts out.

3.2.3 Read latency and write latency fields, MBIR[30:28] and MBIR[33:31]

The read latency and write latency fields of the MBIR are used to specify the read and write latency of the RAM under test. Read and write latencies are the numbers of cycles that the RAM requires to complete read and write operations. For example, in a write to a RAM with a write latency of two cycles, the RAM inputs are valid for a single cycle. The next cycle is a NOP cycle with the chip enable negated. Similarly, in a read from a RAM with a read latency of three cycles, the RAM inputs are valid for a single cycle. After two NOP cycles, the read data is valid on the RAM outputs.

Note

Even if the RAM under test uses the same latency for both read and write operations, you must still program both the read latency and write latency fields of the MBIR with the same value.

Table 3-4 shows the latency settings for read operations.

Table 3-4 Read latency field encoding

Read latency MBIR[30:28]	Number of cycles per read operation
b000	1
b001	2
b010	3
b011	4
b100	5
b101	6
b110	7
b111	8

Table 3-5 shows the latency settings for write operations.

Table 3-5 Write latency field encoding

Write latency MBIR[33:31]	Number of cycles per write operation
b000	1
b001	2
b010	3
b011	4
b100	5
b101	6
b110	7
b111	8

3.2.4 Y-address and X-address fields, MBIR[23:20] and MBIR[27:24]

You can determine the number of address bits you must specify for a RAM from the MBIR fields:

- X-address
- Y-address.

This enables you to specify your address range in two dimensions, this represents the topology of the physical implementation of the RAM more accurately. These two dimensions are controlled by two separate address counters, the X-address counter and the y-address counter. One counter can be incremented or decremented only when the other counter has expired. The chosen test algorithm determines the counter that moves faster.

Use this procedure to determine how many bits to assign to the X-address and Y-address counters:

1. Determine the column width of the RAM array. The Y-address must have at least that many bits for the column select. If it is a Data RAM, then add two bits to that number for the doubleword select.

2. Determine how many address bits the RAM requires. See *L220 Cache Controller RAMs* on page 3-12. Subtract the current Y-address bit number from that number. If the result is eight or fewer bits, then they are all assigned to the X-address for the row select. Otherwise, eight bits are used for the X-address and any unassigned bits are added to the bits already assigned to the Y-address and used for the block select.

Figure 3-2 on page 3-9 shows an example topology for the Data RAM in a 256K level-2 cache.

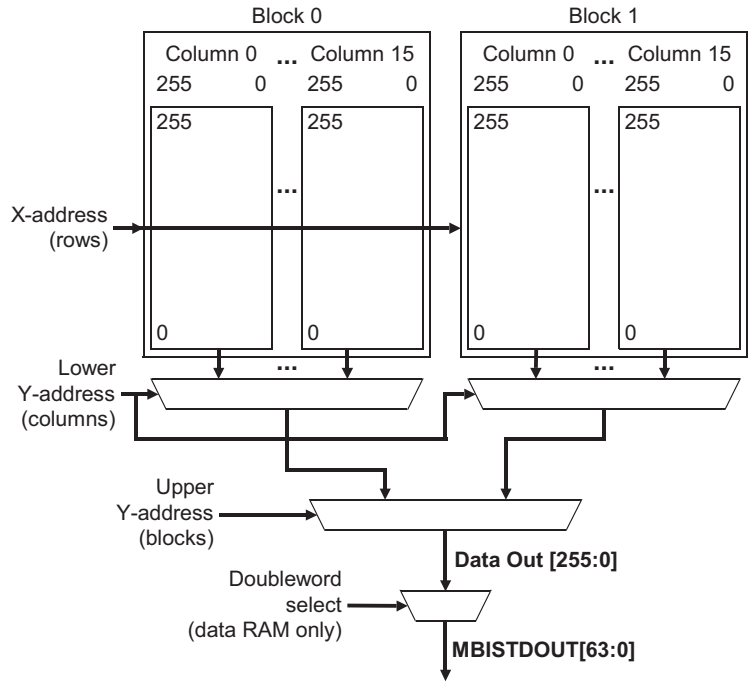


Figure 3-2 Example data RAM topology

The cache RAM in Figure 3-2 has a column width of 16, so it uses four bits for the column address. These four bits map to the least significant bits of the Y-address counter. Because this is a data RAM, it requires two additional doubleword select bits. The doubleword select bits choose between the four 64-bit groups of RAM data before sending the data to the 64-bit **MBISTDOUT[63:0]** bus. These two bits always map to the Y-address counter bits between the column address and the block address.

Because this cache RAM has 256 rows per column, it uses eight bits for the row address, this uses up all eight bits of the X-address counter. This RAM also contains two blocks of 16 columns each, so it uses one bit for the block address. This maps to the most significant bit of the Y-address counter. To correctly test this RAM, the Y-address field must have a value of seven, **MBIR[23:20]** = b0111, and the X-address field must have a value of eight, **MBIR[27:24]** = b1000. Values higher or lower than these produce incorrect results.

Note

If the columns have fewer than 256 rows, you must still assign address bits to the row address until all eight bits are used before assigning any to the block address. If the cache RAM has more than 256 rows per column, then the additional bits must be assigned to the block address. This does not have any detrimental effects on the test coverage of the RAM.

Figure 3-3 shows how the L220 MBIST Controller builds the address output. The doubleword select bits are the least significant two bits of the address. These two bits are ignored unless the data RAM is selected. The exclusive OR of the two least significant bits of the Y-address counter is the least significant bit of the column address for physical addressing of the columns. This is followed by the row address from the X-address counter and, if required, the block address.

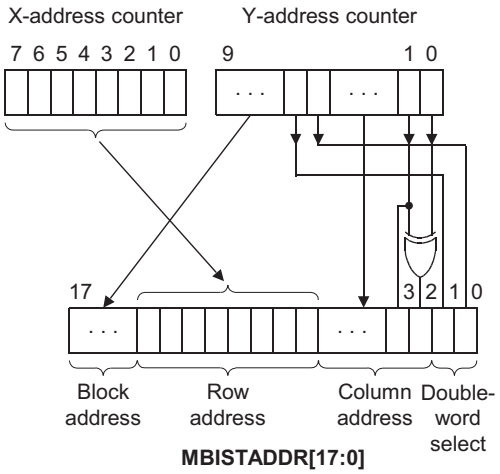


Figure 3-3 MBIST address scrambling

Y-address

The Y-address field specifies the number of Y-address counter bits to use during test. Table 3-6 shows the Y-address settings.

Table 3-6 Y-address field encoding

Y-address MBIR[23:20]	Number of counter bits
<b0010	Unsupported
b0010	2
b0011	3
b0100	4
b0101	5
b0110	6
b0111	7
b1000	8
b1001	9
b1010	10
>b1010	Reserved

X-address

The X-address field specifies the number of X-address counter bits to use during test. Table 3-7 shows the X address settings.

Table 3-7 X-address field encoding

X-address MBIR[27:24]	Number of counter bits
<b0010	Unsupported
b0010	2
b0011	3
b0100	4

Table 3-7 X-address field encoding (continued)

X-address MBIR[27:24]	Number of counter bits
b0101	5
b0110	6
b0111	7
b1000	8
>b1000	Reserved

L220 Cache Controller RAMs

Table 3-8 shows the required sums of the X-address and Y-address fields for complete testing of each RAM type.

Table 3-8 Required sums of X-address and Y-address fields

Cache size	Data RAM	Data parity RAM	Tag or dirty RAMs
128KB	14	12	9
256KB	15	13	10
512KB	16	14	11
1MB	17	15	12
2MB	18	16	13

3.2.5 Data seed field, MBIR[19:16]

The four-bit data seed field supplies the background data for the test algorithm at instruction load.

Note

In the Go/No-Go algorithm, the Read Write Read March (y-fast) and Bang algorithms do not use the data seed value. Table 3-2 on page 3-5 shows the data that the Go/No-Go algorithm uses.

The data seed enables you to select values stored into arrays for $\mathbf{I_{DDQ}}$ ATPG, or to select data words to search for unexpected sensitivities during march or bit-line stress tests. The MBIST engine replicates the four bits of data 16 times to give the full 64 bits of data required on the **MBISTDIN[63:0]** port of the L220 MBIST interface.

3.2.6 Enables field, MBIR[15:5]

Table 3-9 shows how each bit in the enables field selects the cache RAM array to be tested. You can select only one array at a time. Selecting multiple arrays produces unpredictable behavior.

Table 3-9 Enables field encoding

Enables MBIR[15:5]	RAM name
b00000000001	Data
b00000000010	Tag 0
b00000000100	Tag 1
b00000001000	Tag 2
b00000010000	Tag 3
b00000100000	Tag 4
b00001000000	Tag 5
b00010000000	Tag 6
b00100000000	Tag 7
b01000000000	Dirty
b10000000000	Data parity

3.2.7 Column width field, MBIR[4:3]

The column width field specifies the number of columns in each block of RAM in the array under test. The column address is always encoded in the least significant bits of the RAM address, so the number of columns determines the number of bits used. This information is important for the correct operation of certain MBIST operations, such as bit-line stress testing and writing a true physical checkerboard pattern to the array.

Table 3-10 shows the supported column widths along with the number of LSB address bits used for each and the MBIR encodings required to select them.

Table 3-10 Column width field encoding

Column width MBIR[4:3]	Number of columns	Number of address bits
b00	4	2
b01	8	3
b10	16	4
b11	32	5

3.2.8 Cache size field, MBIR[2:0]

The cache size field specifies the size of the cache in your implementation of the L220 module and therefore must always be the same. Table 3-11 shows the supported cache sizes.

Table 3-11 Cache size field encoding

Cache size MBIR[2:0]	Cache size
b000	Reserved
b001	128KB
b010	256KB
b011	512KB
b100	1MB
b101	2MB
b110	Reserved
b111	Reserved

Appendix A
Signal Descriptions

This appendix describes the L220 MBIST Controller signals. It contains the following sections:

A.1 L220 interface signals

Table A-1 shows the L220 MBIST Controller interface signals.

Table A-1 L220 interface

Signal	Type	Description
MBISTDOUT[63:0]	Input	MBIST data out, from L220 **MBISTDOUT[63:0]** = MBIST data out for Data RAM **MBISTDOUT[20:0]** = MBIST data out for Tag RAM **MBISTDOUT[15:0]** = MBIST data out for Dirty RAM **MBISTDOUT[31:0]** = MBIST data out for Data Parity RAM
MBISTADDR[17:0]	Output	MBIST address **MBISTADDR[17:0]** used for Data RAM, two LSBs used as doubleword select **MBISTADDR[14:2]** used for Tag and Dirty RAMs **MBISTADDR[17:2]** used for Data Parity RAM
MBISTCE[10:0]	Output	MBIST RAM chip enables, for writes **MBISTCE[0]** = Data RAM chip enable **MBISTCE[1]** = Tag RAM 0 chip enable **MBISTCE[2]** = Tag RAM 1 chip enable **MBISTCE[3]** = Tag RAM 2 chip enable **MBISTCE[4]** = Tag RAM 3 chip enable **MBISTCE[5]** = Tag RAM 4 chip enable **MBISTCE[6]** = Tag RAM 5 chip enable **MBISTCE[7]** = Tag RAM 6 chip enable **MBISTCE[8]** = Tag RAM 7 chip enable **MBISTCE[9]** = Dirty RAM chip enable **MBISTCE[10]** = Data Parity RAM chip enable

Table A-1 L220 interface (continued)

Signal	Type	Description
MBISTDCTL[12:0]	Output	MBIST control, for reads **MBISTDCTL[1:0]** = MBIST data select for 64 bits of 256 wide data RAM **MBISTDCTL[2]** = MBIST RAM select for Data RAM **MBISTDCTL[3]** = MBIST RAM select for Tag RAM 0 **MBISTDCTL[4]** = MBIST RAM select for Tag RAM 1 **MBISTDCTL[5]** = MBIST RAM select for Tag RAM 2 **MBISTDCTL[6]** = MBIST RAM select for Tag RAM 3 **MBISTDCTL[7]** = MBIST RAM select for Tag RAM 4 **MBISTDCTL[8]** = MBIST RAM select for Tag RAM 5 **MBISTDCTL[9]** = MBIST RAM select for Tag RAM 6 **MBISTDCTL[10]** = MBIST RAM select for Tag RAM 7 **MBISTDCTL[11]** = MBIST RAM select for Dirty RAM **MBISTDCTL[12]** = MBIST RAM select for Data Parity RAM
MBISTDIN[63:0]	Output	MBIST Data In, to L220 **MBISTDIN[63:0]** = MBIST data in for Data RAM **MBISTDIN[20:0]** = MBIST data in for Tag RAM **MBISTDIN[15:0]** = MBIST data in for Dirty RAM **MBISTDIN[31:0]** = MBIST data in for Data Parity RAM
MBISTWE	Output	MBIST Write enable

A.2 Miscellaneous signals

Table A-2 shows the miscellaneous signals.

Table A-2 Miscellaneous signals

Signal	Type	Description
nRESET	Input	Global active LOW reset signal
CLK	Input	Clock
MBISTDATAIN	Input	Serial data in
MBISTDSHIFT	Input	Data log shift
MBISTRESETN	Input	MBIST reset
MBISTRESULT[2:0]	Output	Output status bus
MBISTRUN	Input	Run MBIST test
MBISTSHIFT	Input	Instruction shift
MTESTON	Input	MBIST Mode Enable