ARM L210 MBIST Controller

Copyright © 2003, 2004 ARM Limited. All rights reserved.

Release Information

The table below shows the release state and change history of this document.

<table>
<thead>
<tr>
<th>Date</th>
<th>Issue</th>
<th>Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>27 June, 2003</td>
<td>A</td>
<td>First release</td>
</tr>
<tr>
<td>13 April, 2004</td>
<td>B</td>
<td>r0p1 release</td>
</tr>
<tr>
<td>26 July, 2004</td>
<td>C</td>
<td>Bang pattern erratum.</td>
</tr>
</tbody>
</table>

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
List of Tables


Change history ........................................................................................................ ii
Table 1-1 Signals of the ARM L210 MBIST interface ..................................................... 1-5
Table 2-1 Data log format ............................................................................................... 2-4
Table 3-1 Cache size field encoding ............................................................................... 3-3
Table 3-2 Column width field encoding .......................................................................... 3-4
Table 3-3 Enables field encoding .................................................................................. 3-4
Table 3-4 Y addr field encoding .................................................................................... 3-8
Table 3-5 X addr field encoding .................................................................................. 3-8
Table 3-6 Required sums of x addr and y addr fields .................................................... 3-9
Table 3-7 Read latency field encoding .......................................................................... 3-10
Table 3-8 Write latency field encoding ........................................................................ 3-10
Table 3-9 Control field encoding ................................................................................ 3-11
Table 3-10 Pattern field encoding .............................................................................. 3-12
Table 3-11 Go/No-Go test pattern ............................................................................... 3-14
Table A-1 Signals of the ARM L210 MBIST Controller interface .................................. A-2
List of Figures


Key to timing diagram conventions ................................................................. xi

Figure 1-1  ARM L210 MBIST Controller wiring diagram ................................................................. 1-3
Figure 1-2  Traditional method of interfacing MBIST ................................................................. 1-4
Figure 1-3  MBIST interface of the ARM L210 ................................................................. 1-5
Figure 2-1  Loading the ARM L210 MBIST Controller instruction ................................................................. 2-2
Figure 2-2  Starting the MBIST test ............................................................................................. 2-3
Figure 2-3  Detecting an MBIST failure ....................................................................................... 2-3
Figure 2-4  Start of data log retrieval ............................................................................................. 2-4
Figure 2-5  End of data log retrieval ............................................................................................. 2-4
Figure 2-6  Start of bitmap data log retrieval ................................................................................ 2-6
Figure 2-7  End of bitmap data log retrieval ................................................................................ 2-6
Figure 3-1  Memory BIST Instruction Register ........................................................................... 3-2
Figure 3-2  Example data RAM topology .................................................................................... 3-6
Figure 3-3  MBIST address scrambling ....................................................................................... 3-7
Preface

This preface introduces the *ARM L210 MBIST Controller Technical Reference Manual*. It contains the following sections:

- *About this manual* on page x
- *Feedback* on page xiii.
About this manual

This manual describes the ARM L210 Memory Built-In Self Test (MBIST) Controller.

Product revision status

The rpn identifier indicates the revision status of the product described in this manual, where:
- rn Identifies the major revision of the product.
- pn Identifies the minor revision or modification status of the product.

Intended audience

This manual is written for hardware engineers who are using the ARM L210 MBIST Controller to test the ARM L210 cache.

Using this manual

This manual is organized into the following chapters:

Chapter 1 Introduction

Read this chapter to learn about MBIST technology and the ARM L210 interface to the ARM L210 MBIST Controller.

Chapter 2 Functional Description

Read this chapter to learn 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 to learn how to use the MBIST Instruction Register to configure the mode of operation of the MBIST engine.

Appendix A Signal Descriptions

Read this appendix to learn the names and functions of the ARM L210 MBIST Controller signals.

Conventions

This manual uses the conventions described in:

- Typographical on page xi
- Timing diagrams on page xi
• Signals on page xii.

**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 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*  
Denotes language keywords when used outside example code.

**Timing diagrams**

The figure named *Key to timing diagram conventions* 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.

![Timing Diagram](image)

**Key to timing diagram conventions**

Shading indicates times during which the signal level is undefined. When a signal is shaded, the level of the signal is unimportant and does not affect normal operation.
Signals

The signal conventions are:

Signal level
- Asserted means HIGH for active-HIGH signals and LOW for active-LOW signals.

Suffix N
- Denotes active-LOW in the **MBISTRESE**N signal.

Prefix n
- Denotes active-LOW in the **nRESET** signal.

Further reading

This section lists other relevant publications by ARM Limited.

ARM Limited 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 manual contains information that is specific to the ARM L210 module. See the following documents for other relevant information:

- ARM L210 Technical Reference Manual (ARM DDI 0284)
- ARM L210 Implementation Guide (ARM DII 0069).
Feedback

ARM Limited welcomes feedback on the ARM L210 MBIST Controller and its documentation.

Feedback on the ARM L210 MBIST Controller

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 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 describes the ARM L210 MBIST Controller. It contains the following sections:

- *About the ARM L210 MBIST Controller* on page 1-2
- *ARM L210 MBIST Controller interface* on page 1-3
- *Silicon revision information* on page 1-7.
1.1  About the ARM L210 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. The ARM L210 MBIST Controller is for use with the ARM L210 to perform memory testing of the level-2 cache RAM.

——— Note ——————
This implementation of the ARM L210 MBIST Controller supports only 8-way cache designs.
1.2 ARM L210 MBIST Controller interface

Figure 1-1 shows the ARM L210 MBIST Controller interface to the Automated Test Equipment (ATE) and to the MBIST interface of the ARM L210.

![ARM L210 MBIST Controller wiring diagram](image)

Figure 1-1 ARM L210 MBIST Controller wiring diagram

Figure 1-2 on page 1-4 shows the traditional method of accessing a cache RAM for MBIST.
Because this method significantly reduces the maximum operating frequency, it is not suitable for high-performance designs. Instead, the ARM L210 MBIST Controller uses an additional input to the existing functional multiplexers without reducing maximum operating frequency.

Figure 1-3 on page 1-5 shows the five pipeline stages used to access the cache RAM arrays.
The ARM L210 MBIST Controller accesses memory through the MBIST interface of the ARM L210. Table 1-1 lists the ARM L210 MBIST interface signals.

**Table 1-1 Signals of the ARM L210 MBIST interface**

<table>
<thead>
<tr>
<th>Name</th>
<th>I/O</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>nRESET</td>
<td>Input</td>
<td>Global active LOW reset signal.</td>
</tr>
<tr>
<td>CLK</td>
<td>Input</td>
<td>Active HIGH clock signal. This clock drives the ARM L210 logic.</td>
</tr>
<tr>
<td>MBISTDOUT[63:0]</td>
<td>Output</td>
<td>Data out bus from all cache RAM blocks.</td>
</tr>
</tbody>
</table>

### Table 1-1 Signals of the ARM L210 MBIST interface (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>I/O</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MBISTDCTL[12:0]</td>
<td>Input</td>
<td>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. ( \text{MBISTDCTL[12:0]} = \text{delayed {MBISTCE[10:0], MBISTADDR[1:0]}} ).</td>
</tr>
<tr>
<td>MTESTON</td>
<td>Input</td>
<td>Select signal for cache RAM array. This signal is the select input to the multiplexers that access the cache RAM arrays for test. When asserted, MTESTON takes priority over all other select inputs to the multiplexers.</td>
</tr>
<tr>
<td>MBISTCE[10:0]</td>
<td>Input</td>
<td>One-hot chip enable signals to select cache RAM arrays for test.</td>
</tr>
<tr>
<td>MBISTWE</td>
<td>Input</td>
<td>Global write enable signal for all RAM arrays.</td>
</tr>
<tr>
<td>MBISTADDR[17:0]</td>
<td>Input</td>
<td>Address signal for cache RAM array. MBISTADDR[1:0] is the doubleword select value. See Y addr and xaddr fields, MBIR[23:20] and MBIR[27:24] on page 3-5 for a description of the doubleword select. Not all RAM arrays use the full address width.</td>
</tr>
<tr>
<td>MBISTDIN[63:0]</td>
<td>Input</td>
<td>Data bus to the cache RAM arrays. Not all RAM arrays use the full data width.</td>
</tr>
</tbody>
</table>
1.3 Silicon revision information

This manual is for revision r0p1 of the L210 MBIST Controller. See Product revision status on page x for details of revision numbering.

The r0p1 L2 MBIST Controller is updated to correct errata in the r0p0 L2 MBIST Controller. There are no other functional differences between the r0p0 and r0p1 L2 MBIST Controllers.
Chapter 2
Functional Description

This chapter provides timing sequences for loading instructions, starting the MBIST engine, detecting failures, and retrieving the data log. It contains the following sections:

- Timing on page 2-2
- Bitmap mode on page 2-6.
2.1 Timing

A 46-bit instruction, loaded serially at the start of each test, controls the operation of the ARM L210 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 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 ARM L210 MBIST Controller:
- Instruction load
- Starting MBIST
- Failure detection on page 2-3
- Data log retrieval on page 2-3.

2.1.1 Instruction load

To load an MBIST instruction, drive MBISTSHIFT HIGH. At the next rising clock edge, the 46-bit shift sequence begins as Figure 2-1 shows. To enable data input from the ATE, the PLL is in bypass mode, and the clock is not running at test frequency.

![Figure 2-1 Loading the ARM L210 MBIST Controller instruction](image)

2.1.2 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 Figure 2-2 on page 2-3 shows.
2.1.3 Failure detection

The MBISTRESULT[1] flag goes HIGH two CLK cycles after the controller detects a failure, as Figure 2-3 shows. It stays HIGH if sticky fail is enabled. If stop on fail is enabled, the MBISTRESULT[2] flag goes HIGH two cycles later.

Note
To ensure that a failure at test speed can be observed by the ATE, specify a sticky fail in the MBIST instruction. See Control field, MBIR[39:34] on page 3-11.

2.1.4 Data log retrieval

During a test, the ARM L210 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 2-4 on page 2-4 and Figure 2-5 on page 2-4 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 2-4 on page 2-4 shows. To begin shifting out the data log on MBISTRESULT[0], drive MBISTSHIFT HIGH. The

---

Figure 2-2 Starting the MBIST test

Figure 2-3 Detecting an MBIST failure
**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 2-4 Start of data log retrieval](image)

When the last data log bit shifts out, drive **MBISTDSHIFT** LOW as Figure 2-5 shows.

![Figure 2-5 End of data log retrieval](image)

Table 2-1 shows the format of the data log.

**Table 2-1 Data log format**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[85:68]</td>
<td>Address of the failing location.</td>
</tr>
<tr>
<td>[67:4]</td>
<td>Failing data bits. These bits are set for faulty bits and cleared for passing bits.</td>
</tr>
<tr>
<td>[3:0]</td>
<td>The data seed used in the test. See Data seed field, <strong>MBIR[19:16]</strong> on page 3-5.</td>
</tr>
</tbody>
</table>

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 ARM L210. It always includes the doubleword select value in the least significant two bits (see **Y**...
addr and xaddr fields, MBIR[23:20] and MBIR[27:24] on page 3-5 for more information on the doubleword select value). Contact ARM if you require more information.
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 2-6 shows.

![Figure 2-6 Start of bitmap data log retrieval](image)

After you finish shifting and drive MBISTDSHIFT LOW, the controller then resumes testing where it stopped as Figure 2-7 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 2-7 End of bitmap data log retrieval](image)

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 MBIST engine. It contains the following sections:

- *About the MBIST Instruction Register* on page 3-2
- *Field descriptions* on page 3-3.
3.1 About the MBIST Instruction Register

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

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

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

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

**Enables** Specifies the RAM under test.

**Data seed** Specifies the four-bit data background.

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

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

**Read latency** Specifies the number of cycles to allow for a RAM read.

**Write latency** Specifies the number of cycles to allow for a RAM write.

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

**Pattern** Specifies the test algorithm.

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

The following sections describe the MBIR fields:
- Cache size field, MBIR[2:0]
- Column width field, MBIR[4:3] on page 3-4
- Enables field, MBIR[15:5] on page 3-4
- Data seed field, MBIR[19:16] on page 3-5
- Read latency and write latency fields, MBIR[30:28] and MBIR[33:31] on page 3-9
- Control field, MBIR[39:34] on page 3-11
- Pattern field, MBIR[45:40] on page 3-11.

3.2.1 Cache size field, MBIR[2:0]

The cache size field specifies the size of the cache in your implementation of the ARM L210 module and therefore must always be the same. Table 3-1 shows the supported cache sizes.

<table>
<thead>
<tr>
<th>Cache size MBIR[2:0]</th>
<th>Cache size</th>
</tr>
</thead>
<tbody>
<tr>
<td>b000</td>
<td>Reserved</td>
</tr>
<tr>
<td>b010</td>
<td>128K</td>
</tr>
<tr>
<td>b011</td>
<td>256K</td>
</tr>
<tr>
<td>b100</td>
<td>512K</td>
</tr>
<tr>
<td>b101</td>
<td>1M</td>
</tr>
<tr>
<td>b110</td>
<td>2M</td>
</tr>
<tr>
<td>b111</td>
<td>Reserved</td>
</tr>
</tbody>
</table>
3.2.2 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 3-2 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>
<thead>
<tr>
<th>Column width MBIR[4:3]</th>
<th>Number of columns</th>
<th>Number of address bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>b00</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>b01</td>
<td>8</td>
<td>3</td>
</tr>
<tr>
<td>b10</td>
<td>16</td>
<td>4</td>
</tr>
<tr>
<td>b11</td>
<td>32</td>
<td>5</td>
</tr>
</tbody>
</table>

3.2.3 Enables field, MBIR[15:5]

Table 3-3 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 Undefined behavior.

<table>
<thead>
<tr>
<th>Enables MBIR[15:5]</th>
<th>RAM name</th>
</tr>
</thead>
<tbody>
<tr>
<td>b000000000001</td>
<td>Data</td>
</tr>
<tr>
<td>b000000000010</td>
<td>Tag 0</td>
</tr>
<tr>
<td>b000000000100</td>
<td>Tag 1</td>
</tr>
<tr>
<td>b000000010000</td>
<td>Tag 2</td>
</tr>
<tr>
<td>b000000100000</td>
<td>Tag 3</td>
</tr>
<tr>
<td>b000001000000</td>
<td>Tag 4</td>
</tr>
<tr>
<td>b000010000000</td>
<td>Tag 5</td>
</tr>
</tbody>
</table>
3.2.4 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. See Table 3-11 on page 3-14 for the data used in the Go/No-Go algorithm.

The data seed enables you to select values stored into arrays for $I_{DDQ}$ 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 ARM L210 MBIST interface.

3.2.5 Y addr and xaddr fields, MBIR[23:20] and MBIR[27:24]

The number of address bits you must specify for a RAM can be determined from the MBIR fields:

- x addr
- y addr.

This enables you to specify your address range in two dimensions, which 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 which counter 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 ARM L210 RAMs on page 3-9). 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 is an example topology for the data RAM in a 256K level-2 cache.

![Figure 3-2 Example data RAM topology](image)

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, which 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, which maps to the most significant bit of the y-address counter. To correctly test this RAM, the y addr field must have a value of seven (**MBIR[23:20]** = b0111), and the X addr 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 ARM L210 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 addr

The y addr field specifies the number of y-address counter bits to use during test. Table 3-4 lists the y addr settings.

<table>
<thead>
<tr>
<th>Y addr MBIR[23:0]</th>
<th>Number of counter bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;b0010</td>
<td>Unsupported</td>
</tr>
<tr>
<td>b0010</td>
<td>2</td>
</tr>
<tr>
<td>b0011</td>
<td>3</td>
</tr>
<tr>
<td>b0100</td>
<td>4</td>
</tr>
<tr>
<td>b0101</td>
<td>5</td>
</tr>
<tr>
<td>b0110</td>
<td>6</td>
</tr>
<tr>
<td>b0111</td>
<td>7</td>
</tr>
<tr>
<td>b1000</td>
<td>8</td>
</tr>
<tr>
<td>b1001</td>
<td>9</td>
</tr>
<tr>
<td>b1010</td>
<td>10</td>
</tr>
<tr>
<td>&gt;b1010</td>
<td>Reserved</td>
</tr>
</tbody>
</table>

X addr

The x addr field specifies the number of x-address counter bits to use during test. Table 3-5 lists the x addr settings.

<table>
<thead>
<tr>
<th>X addr MBIR[27:24]</th>
<th>Number of counter bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;b0010</td>
<td>Unsupported</td>
</tr>
<tr>
<td>b0010</td>
<td>2</td>
</tr>
<tr>
<td>b0011</td>
<td>3</td>
</tr>
<tr>
<td>b0100</td>
<td>4</td>
</tr>
</tbody>
</table>
Table 3-5 X addr field encoding (continued)

<table>
<thead>
<tr>
<th>X addr MBIR[27:24]</th>
<th>Number of counter bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>b0101</td>
<td>5</td>
</tr>
<tr>
<td>b0110</td>
<td>6</td>
</tr>
<tr>
<td>b0111</td>
<td>7</td>
</tr>
<tr>
<td>b1000</td>
<td>8</td>
</tr>
<tr>
<td>&gt;b1000</td>
<td>Reserved</td>
</tr>
</tbody>
</table>

**ARM L210 RAMs**

Table 3-6 shows the required sums of the x addr and y addr fields for complete testing of each RAM type.

Table 3-6 Required sums of x addr and y addr fields

<table>
<thead>
<tr>
<th>Cache size</th>
<th>Data RAM</th>
<th>Data parity RAM</th>
<th>Tag or dirty RAMs</th>
</tr>
</thead>
<tbody>
<tr>
<td>128K</td>
<td>14</td>
<td>12</td>
<td>9</td>
</tr>
<tr>
<td>256K</td>
<td>15</td>
<td>13</td>
<td>10</td>
</tr>
<tr>
<td>512K</td>
<td>16</td>
<td>14</td>
<td>11</td>
</tr>
<tr>
<td>1M</td>
<td>17</td>
<td>15</td>
<td>12</td>
</tr>
<tr>
<td>2M</td>
<td>18</td>
<td>16</td>
<td>13</td>
</tr>
</tbody>
</table>

**3.2.6 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 3-7 shows the latency settings for read operations.

<table>
<thead>
<tr>
<th>Read latency MBIR[30:28]</th>
<th>Number of cycles per read operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>b000</td>
<td>1</td>
</tr>
<tr>
<td>b001</td>
<td>2</td>
</tr>
<tr>
<td>b010</td>
<td>3</td>
</tr>
<tr>
<td>b011</td>
<td>4</td>
</tr>
<tr>
<td>b100</td>
<td>5</td>
</tr>
<tr>
<td>b101</td>
<td>6</td>
</tr>
<tr>
<td>b110</td>
<td>7</td>
</tr>
<tr>
<td>b111</td>
<td>8</td>
</tr>
</tbody>
</table>

Table 3-8 shows the latency settings for write operations.

<table>
<thead>
<tr>
<th>Write latency MBIR[33:31]</th>
<th>Number of cycles per write operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>b000</td>
<td>1</td>
</tr>
<tr>
<td>b001</td>
<td>2</td>
</tr>
<tr>
<td>b010</td>
<td>3</td>
</tr>
<tr>
<td>b011</td>
<td>4</td>
</tr>
<tr>
<td>b100</td>
<td>5</td>
</tr>
</tbody>
</table>
3.2.7 Control field, MBIR[39:34]

The control field specifies the MBIST function. Table 3-9 shows how the control field affects the behavior of the ARM L210 MBIST Controller.

Table 3-9 Control field encoding

<table>
<thead>
<tr>
<th>Control MBIR[39:34]</th>
<th>Behavior</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bx00000</td>
<td>Default</td>
<td>Test runs to completion. If MBIR[39] = 0, sticky fail present after first failure.</td>
</tr>
<tr>
<td>bx00001</td>
<td>Stop on fail</td>
<td>End of test on failure.</td>
</tr>
<tr>
<td>bx00011</td>
<td>Bitmap mode</td>
<td>Enables logging of each failure. See Bitmap mode on page 2-6.</td>
</tr>
</tbody>
</table>

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.

3.2.8 Pattern field, MBIR[45:40]

The ARM L210 MBIST controller comes 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 3-10 describes the supported algorithms, and *Pattern specification* describes their use. The N values in the table refer to the number of RAM accesses per address location and give an indication of the test time when using that algorithm.

**Table 3-10 Pattern field encoding**

<table>
<thead>
<tr>
<th>Pattern MBIR[45:40]</th>
<th>Algorithm name</th>
<th>N</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>000000</td>
<td>Write Solids</td>
<td>1N</td>
<td>Write a solid pattern to memory</td>
</tr>
<tr>
<td>000001</td>
<td>Read Solids</td>
<td>1N</td>
<td>Read a solid pattern from memory</td>
</tr>
<tr>
<td>000010</td>
<td>Write Checkerboard</td>
<td>1N</td>
<td>Write a checkerboard pattern to memory</td>
</tr>
<tr>
<td>000011</td>
<td>Read Checkerboard</td>
<td>1N</td>
<td>Read a checkerboard pattern from memory</td>
</tr>
<tr>
<td>000100</td>
<td>March C+ (x-fast)</td>
<td>14N</td>
<td>March C+ algorithm, incrementing x-address first</td>
</tr>
<tr>
<td>000101</td>
<td>March C+ (y-fast)</td>
<td>14N</td>
<td>March C+ algorithm, incrementing y-address first</td>
</tr>
<tr>
<td>000101</td>
<td>Fail Pattern</td>
<td>6N</td>
<td>Tests memory failure detection capability</td>
</tr>
<tr>
<td>b001010</td>
<td>Read Write March (x-fast)</td>
<td>6N</td>
<td>Read write march pattern, incrementing x-address first</td>
</tr>
<tr>
<td>b001011</td>
<td>Read Write March (y-fast)</td>
<td>6N</td>
<td>Read write march pattern, incrementing y-address first</td>
</tr>
<tr>
<td>b010000</td>
<td>Read Write Read March (x-fast)</td>
<td>8N</td>
<td>Read write read march pattern, incrementing x-address first</td>
</tr>
<tr>
<td>b010001</td>
<td>Read Write Read March (y-fast)</td>
<td>8N</td>
<td>Read write read march pattern, incrementing y-address first</td>
</tr>
<tr>
<td>b010100</td>
<td>Bang</td>
<td>18N</td>
<td>Bit-line stress pattern</td>
</tr>
<tr>
<td>b111111</td>
<td>Go/No-Go</td>
<td>30N</td>
<td>See Table 3-11 on page 3-14</td>
</tr>
</tbody>
</table>

**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 addr and xaddr fields, MBIR[23:20] and MBIR[27:24]* on page 3-5 describes the x-address and y-address counters.

The first four patterns are useful for data retention or IDDQ 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 is used to describe the algorithm:

- 0 represents the data seed
- 1 represents the inverse data seed
- w represents a write operation
- r represents a read operation
- \( \uparrow \) indicates that the address is incremented
- \( \downarrow \) indicates that the address is decremented.

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

This is the industry-standard March C+ algorithm:

\[ \uparrow (w0) \uparrow (r0, w1, r1) \uparrow (r1, w0, r0) \downarrow (r0, w1, r1) \downarrow (r1, w0, r0) \uparrow (r0) \]

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

\[ \uparrow (w0) \uparrow (r0, w1) \downarrow (r1, w0) \uparrow (r0) \]

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

\[ \uparrow (w0) \uparrow (r0, w1, r1) \downarrow (r1, w0, r0) \uparrow r0 \]

Bang

This test is always performed 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, which is always the first row of the column being addressed.

\[ \uparrow (w0) \uparrow (r0, w1, w1(row 0)x6) \uparrow (r1x5, w0(row 0), r1, w0) \uparrow (r0) \]
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 shown in Table 3-11.

Table 3-11 Go/No-Go test pattern

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Algorithm</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Write Checkerboard</td>
<td>Data seed</td>
</tr>
<tr>
<td>2</td>
<td>Read Checkerboard</td>
<td>Data seed</td>
</tr>
<tr>
<td>3</td>
<td>Write Checkerboard</td>
<td>NOT(Data seed)</td>
</tr>
<tr>
<td>4</td>
<td>Read Checkerboard</td>
<td>NOT(Data seed)</td>
</tr>
<tr>
<td>5</td>
<td>Read Write Read March (y-fast)</td>
<td>0x6</td>
</tr>
<tr>
<td>6</td>
<td>Bang</td>
<td>0xF</td>
</tr>
</tbody>
</table>

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.
Appendix A

Signal Descriptions

This appendix describes the ARM L210 MBIST Controller signals. It contains the following section:

- Signal descriptions on page A-2.
# A.1 Signal descriptions

Table A-1 lists the ARM L210 MBIST Controller interface signals.

<table>
<thead>
<tr>
<th>Pin</th>
<th>I/O</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLK</td>
<td>Input</td>
<td>Active HIGH clock signal. This clock drives the ARM L210 MBIST Controller logic.</td>
</tr>
<tr>
<td>MBISTDOUT[63:0]</td>
<td>Input</td>
<td>Data out bus from all RAM arrays.</td>
</tr>
<tr>
<td>MBISTDCTL[12:0]</td>
<td>Output</td>
<td>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]}.</td>
</tr>
<tr>
<td>MBISTRESETN</td>
<td>Input</td>
<td>Active LOW reset signal for the ARM L210 MBIST Controller logic. This signal must be asserted for at least one full CLK cycle before programming the controller for the first test.</td>
</tr>
<tr>
<td>MTESTON</td>
<td>Input</td>
<td>RAM array select signal. This signal is the select input to the multiplexers that access the RAM arrays for test. MTESTON also enables the registers in the L210 MBIST Controller to clock in new data.</td>
</tr>
<tr>
<td>MBISTDSHIFT</td>
<td>Input</td>
<td>Shift out. Asserting this signal enables serial unload of the data log.</td>
</tr>
<tr>
<td>MBISTRUN</td>
<td>Input</td>
<td>Test run. Asserting this signal begins the programmed test algorithm.</td>
</tr>
<tr>
<td>MBISTSHIFT</td>
<td>Input</td>
<td>Shift in. Asserting this signal enables serial load of the MBIST Instruction Register.</td>
</tr>
<tr>
<td>MBISTDATAIN</td>
<td>Output</td>
<td>Serial data input for loading the MBIST Instruction Register.</td>
</tr>
</tbody>
</table>
| MBISTRESULT[2:0]     | Output| ARM L210 MBIST Controller output. This signal provides test status and serial data log output:  
|                      |      | MBISTRESULT[2] = test complete flag, asserted at the end of the test        |
|                      |      | MBISTRESULT[1] = fail flag, asserted when failure is detected               |
|                      |      | MBISTRESULT[0] = address expire flag or data log output. During testing, this signal goes HIGH each time both the x-address and y-address counters expire, which indicates progression to the next stage of the selected test pattern. When shifting out the data log, this signal is the serial data output. |
| MBISTCE[10:0]        | Output| One-hot chip enable signals to select RAM blocks for test.                 |
Table A-1 Signals of the ARM L210 MBIST Controller interface (continued)

<table>
<thead>
<tr>
<th>Pin</th>
<th>I/O</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MBISTWE</td>
<td>Output</td>
<td>Global write enable signal for all RAM arrays.</td>
</tr>
<tr>
<td>MBISTADDR[17:0]</td>
<td>Output</td>
<td>RAM array address signal. <strong>MBISTADDR[1:0]</strong> is the doubleword select value. See <em>Y addr and xaddr fields, MBIR[23:20] and MBIR[27:24]</em> on page 3-5 for a description of the doubleword select. Not all RAM arrays use the full address width.</td>
</tr>
<tr>
<td>MBISTDIN[63:0]</td>
<td>Output</td>
<td>Data bus to the RAM arrays. Not all RAM arrays use the full data width.</td>
</tr>
</tbody>
</table>
Index

A
Address expire flag A-2
Address scrambling 3-7
Analog testing 3-13
see also Bit-line stress test
ARM L210
MBIST interface 1-3
ARM L210 MBIST interface signals 1-5
ATE
clock 2-2
data input 2-2
failure observation 2-3
interface 1-3, 1-6
test frequency 3-11
Automated test equipment
see ATE

B
Background data

C
Cache RAM
array address signal 1-6
array select signal 1-6
array write enable 1-6
chip enables 1-6
data bus 1-6
Cache size field 3-2
encoding 3-3
CLK 1-5, A-2
Column width field 3-2
encoding 3-4
Control field 3-2

see Data seed
Bang algorithm 3-5, 3-12, 3-13
Bit-line stress test 3-4, 3-5, 3-11, 3-12, 3-13
Bitmap mode 2-6, 3-11
Block select 3-6, 3-7
Bypass mode 2-2
clock 1-5, A-2

D
Data log
bitmap mode 2-6
doubleword select value 2-4
format 2-4
output 2-3
retrieval 2-3
Data parity RAM 3-5, 3-9
Data RAM 3-4
column width 3-6
doubleword select 3-6, 3-7
Data seed 2-4, 3-5, 3-12, 3-13, 3-14
Data seed field 3-2
Dirty RAM 3-5, 3-9
Doubleword select 1-6, 2-4, 3-6, 3-7
Index

E
Enables field 3-2
   encoding 3-4

F
Fail flag 2-3, 3-2, 3-11, A-2
   timing 2-3
Fail Pattern algorithm 3-12

G
Go/No-Go algorithm 3-5, 3-12

L
Level-2 cache RAM
   testing 1-2
   topology 3-6

M
March C+ (x-fast) algorithm 3-12, 3-13
March C+ (y-fast) algorithm 3-12, 3-13
MBIR 3-1
   instruction format 3-2
   encoding 3-3
MBIST instruction 2-3
   format 3-2
   loading 2-2
MBIST Instruction Register
   see MBIR
MBISTADDR 1-6, A-3
MBISTCE 1-6, A-2
MBISTDATAIN A-2
MBISTDCTL 1-6
MBISTDIN 1-6, A-3
MBISTDOUT 1-5
MBISTDSHIFT A-2
MBISTRESETN A-2
MBISTRUN A-2
MBISTSHIFT A-2
MBISTWE 1-6, A-3
MTESTON 1-6, A-2

N
nRESET 1-5

O
On-chip PLL 2-2
   bypass mode 2-2

P
Pattern field 3-2
   encoding 3-12

R
RAM array enables 3-2
Read Checkerboard algorithm 3-12, 3-13
Read latency field 3-2, 3-9
   encoding 3-10
Read Solids algorithm 3-12
Read Write March (x-fast) algorithm 3-12, 3-13
Read Write March (y-fast) algorithm 3-12
Read Write March (x-fast) algorithm 3-12
Read Write March (y-fast) algorithm 3-5, 3-12, 3-13
Read Write March (y-fast) algorithm 3-13
Read Write Read March (x-fast) algorithm 3-12
Read Write Read March (y-fast) algorithm 3-12
Read Write Read March (y-fast) algorithm 3-12
Read Write Read March (y-fast) algorithm 3-13

S
Sticky fail flag 2-3, 3-2, 3-11
   timing 2-3
Stop on fail 3-11

T
Tag RAM 3-9
Test algorithms 1-2, 3-5
Test complete flag 2-3, 2-4, A-2

W
Write Checkerboard algorithm 3-12, 3-13
Write latency field 3-2, 3-9
   encoding 3-10
Write Solids algorithm 3-12

X
X addr field 3-2, 3-8
   encoding 3-8
X-address counter 3-2, 3-7, 3-8
determining size 3-5

Y
Y addr field 3-2, 3-7, 3-8
   encoding 3-8
Y-address counter 3-2, 3-7, 3-8
determining size 3-5