CoreSight™ ETM11™

Revision: r1p1

Technical Reference Manual

ARM®

ARM DDI 0318E

CoreSight ETM11
Technical Reference Manual

Release Information

The following changes have been made to this book.

Change History

Date	Issue	Confidentiality	Change
19 July 2004	A	Non-Confidential	First release.
13 May 2005	B	Non-Confidential	New issue for r0p1 release. Minor updates and corrections throughout document.
10 October 2006	C	Non-Confidential	New issue for r0p2 release. Minor updates and corrections throughout document.
08 December 2006	D	Non-Confidential	New issue for r1p0 release.
17 May 2007	E	Non-Confidential	Updated for r1p1. Appendix A revised. Appendix B *Intelligent Energy Management (IEM) System Functionality* moved to *CoreSight ETM11 Integration Manual*.

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
CoreSight ETM11 Technical Reference Manual

List of Tables
CoreSight ETM11 Technical Reference Manual

List of Figures
CoreSight ETM11 Technical Reference Manual

Preface

This preface introduces the *CoreSight™ ETM11™ Technical Reference Manual*. It contains the following sections:

- *About this manual* on page xii
- *Feedback* on page xvi.

About this manual

This is the *Technical Reference Manual* (TRM) for CoreSight ETM11.

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 the following target audiences:

- Designers of development tools providing support for ETM functionality. Implementation-specific behavior is described in this document. These users can also consult the *ETM Architecture Specification* (ARM IHI 0014).
- Hardware and software engineers integrating the CoreSight ETM11 into an ASIC that includes an ARM11 processor. These users can also consult the *CoreSight ETM11 Integration Manual* (ARM DII 0098).

Using this manual

This manual is organized into the following chapters:

Chapter 1 *Introduction*

Read this chapter for an introduction to the functionality of CoreSight ETM11.

Chapter 2 *Implementation-defined Behavior*

Read this chapter for a description of the implementation-defined features of CoreSight ETM11.

Chapter 3 *Programmer's Model*

Read this chapter for a description of the programmer's model for CoreSight ETM11.

Chapter 4 *Blocks for Stand-alone CoreSight ETM11*

CoreSight ETM11 can be used stand-alone, that is, without a CoreSight subsystem, to trace a single ARM11 family processor. This chapter describes the additional blocks provided for CoreSight ETM11 used stand-alone.

Appendix A *Signals Lists*

Read this appendix for a description of all CoreSight ETM11 signals.

Appendix B *I/O Signal Timings*

Read this appendix for a description of the CoreSight ETM11 timing parameters.

Glossary Read the Glossary for definitions of terms used in this manual.

Conventions

Conventions that this manual can use are described in:

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

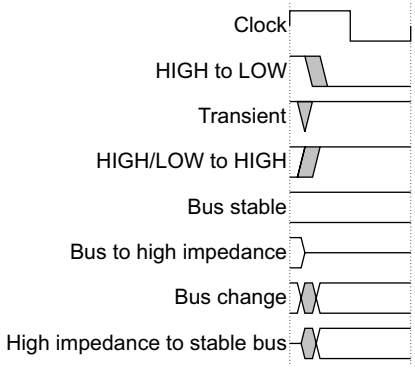
Typographical

This manual uses the following typographical conventions:

italic	Highlights important notes, introduces special terminology, denotes internal cross-references, and citations.
bold	Highlights interface elements, such as menu names. Denotes 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. The replaceable terms appear in normal font in running text. For example: • `MRC p15, 0 <Rd>, <CRn>, <CRm>, <Opcode_2>` • The Opcode_2 value selects which register is accessed.

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. Shaded bus and signal areas are undefined, and the bus or signal can assume any value within the shaded area at that time. The actual level is unimportant and does not affect normal operation.



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.

Lower-case n Denotes an active-LOW signal.

Prefix A Denotes global *Advanced eXtensible Interface* (AXI) signals:

Prefix AR Denotes AXI read address channel signals.

Prefix AW Denotes AXI write address channel signals.

Prefix B Denotes AXI write response channel signals.

Prefix C Denotes AXI low-power interface signals.

Prefix H Denotes *Advanced High-performance Bus* (AHB) signals.

Prefix P Denotes *Advanced Peripheral Bus (*APB) signals.

Prefix R Denotes AXI read data channel signals.

Prefix W Denotes AXI write data 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 Frequently Asked Questions list.

ARM publications

This manual contains information that is specific to the CoreSight ETM11. See the following documents for other relevant information:

- *AMBA 3 APB Protocol Specification (*ARM IHI 0024)
- *CoreSight ETM11 Configuration and Sign-off Guide (*ARM DII 0097)
- *CoreSight ETM11 Integration Manual* (ARM DII 0187)
- *ETM Architecture Specification* (ARM IHI 0014)
- *CoreSight Architecture v1.0 Specification* (ARM IHI 0029)
- *CoreSight Components Technical Reference Manual* (ARM DDI 0314)
- *Intelligent Energy Controller r0p0 Technical Overview* (ARM DTO 0005).

In addition, see the following documentation for specific information relating to ARM products:

- *ARM Reference Peripheral Specification* (ARM DDI 0062)
- the ARM datasheet or technical reference manual for the core to which the ETM11 macrocell is to be connected.

Feedback

ARM Limited welcomes feedback on the CoreSight ETM11 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 your emails to `errata@arm.com` giving:

- the document title
- the document 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 introduces the CoreSight ETM11 macrocell. It contains the following sections:

1.1 About CoreSight ETM11

CoreSight ETM11 provides instruction trace and data trace for the ARM11 family of microprocessors. CoreSight ETM11 supports all current ARM11 cores and is compatible with TrustZone and Thumb-2.

You can use the CoreSight ETM11 macrocell in two contexts:

- Stand-alone, as the ETM11CSSingle module. You use the ETM11CSSingle module to trace only a single ARM CPU, for example an ARM1176 processor. See Chapter 4 *Blocks for Stand-alone CoreSight ETM11* for information about using a stand-alone CoreSight ETM11.
- In a CoreSight System, as the ETM11CS module. See the *CoreSight Components Technical Reference Manual* for information about using CoreSight ETM11 in a CoreSight system.

Figure 1-1 shows the main functional blocks and clock domains of CoreSight ETM11.

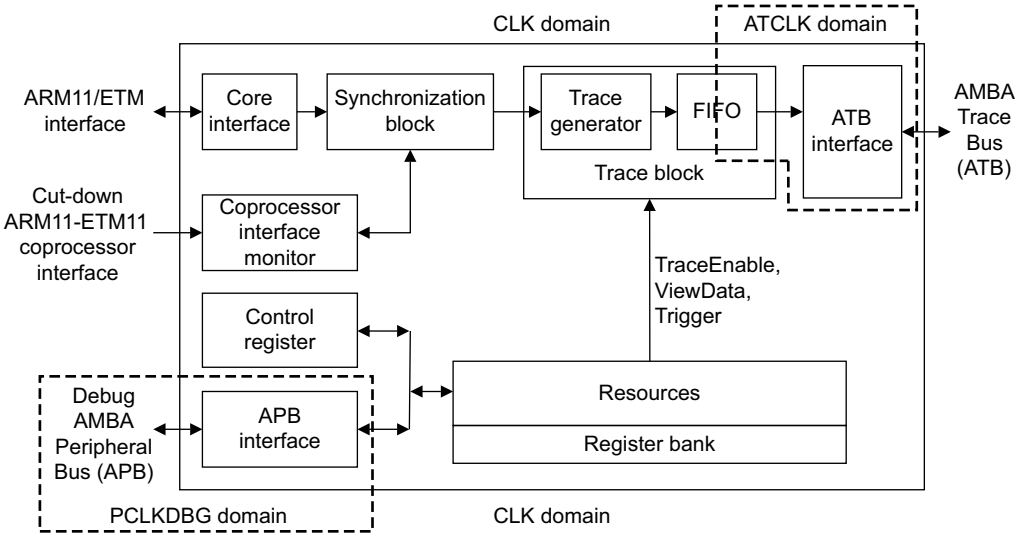


Figure 1-1 CoreSight ETM11 functional blocks and clock domains

Note

The signal connections for an ARM11 MPCore processor are different to those for other ARM11 family processors. For more information on these signals, see Appendix A *Signals Lists* and Appendix B *I/O Signal Timings*.

See the *Embedded Trace Macrocell Architecture Specification* for information about the trace protocol, and about controlling tracing using triggering and filtering resources, and ETM sharing.

Programming access to CoreSight ETM11 is through the APB interface:

- In a CoreSight system, the APB interface connects to the Debug APB bus.
- In a stand-alone CoreSight ETM11 the APB interface connects to the ETMJTAGPORT. For information about the ETMJTAGPORT see *ETMJTAGPORT* on page 4-3.

CoreSight ETM11 is *Intelligent Energy Management* (IEM)-ready. The RTL can be configured to include the required interface to permit the CoreSight ETM11 to be deployed in an IEM capable system.

For more information about typical APB transfers on the APB interface, see the *AMBA 3 APB Protocol Specification.*

1.2 CoreSight ETM11 configuration

Table 1-1 shows the CoreSight ETM11 resources.

Table 1-1 CoreSight ETM11 resources

Resource description	Number/size
Context ID comparators	1
External inputs	4
External outputs	2
Sequencers	1
Counters	2
Memory map decoders	0
Data comparators	2
Pairs of address comparators	4
Extended external inputs	20
Extended external input selectors	2
FIFO size	72 bytes
Trace port configuration	Variable

1.3 Product revisions

This is the Technical Reference Manual for CoreSight ETM11 r1p1. This section summarizes the differences in functionality between the releases of the CoreSight ETM11 macrocell.

r0p0 - r0p1 Incorporates engineering errata fixes and contains the following differences in functionality:

- The **PSLVERRDBG** output signal now indicates an error if you access a register on an APB interface that is in a powered-down domain of the CoreSight ETM11. On the r0p0 release the **PSLVERRDBG** signal is tied LOW.
- All state is lost when the core domain is powered down. Therefore, a reset is applied on power up of the core domain. After this reset, the FIFO read and write pointers are both reset to zero. On the r0p0 release only the FIFO write pointer is reset to zero.

r0p1 - r0p2 Incorporates engineering errata fixes. It contains no differences in functionality.

r0p2 - r1p0 Contains additional signals **ETMDD2[63:0]** and **ETMDDCTL2[1:0]** to support the tracing of the ARM11 MPCore processor.

r1p0 - r1p1 Incorporates engineering errata fixes. It contains no differences in functionality.

Chapter 2
Implementation-defined Behavior

This chapter contains implementation-specific information relating to the CoreSight ETM11. It contains the following sections:

2.1 ETM architecture version

CoreSight ETM11 implements version 3.2 of the ETM architecture, ETMv3.2. See the *ETM Architecture Specification* for more information.

2.2 CoreSight ETM11 registers summary

Table 2-1 lists the CoreSight ETM11 registers.

Table 2-1 CoreSight ETM11 registers summary

Name	Base offset	Type	Reset value	Description
Configuration Code	0x004	RO	0x8D294024	See *Configuration Code Register* on page 2-7
ETM ID	0x1E4	RO	0x410n3225[a]	See *ETM ID Register* on page 2-5
Configuration Code Extension	0x1E8	RO	0x000008A2	See *Configuration Code Extension Register* on page 2-8
ITMISCOUT	0xEDC	WO	-	See *ITMISCOUT Register (miscellaneous outputs)* on page 2-14
ITMISCIN	0xEE0	RO	-[b]	See *ITMISCIN Register (miscellaneous inputs)* on page 2-15
ITTRIGGERACK	0xEE4	RO	-[b]	See *ITTRIGGERACK Register (trigger acknowledge)* on page 2-16
ITTRIGGERREQ	0xEE8	WO	-	See *ITTRIGGERREQ Register (trigger request)* on page 2-16
ITATBDATA0	0xEEC	WO	-	See *ITATBDATA0 Register (ATB data 0)* on page 2-17
ITATBCTR2	0xEF0	RO	-[b]	See *ITATBCTR2 Register (ATB control 2)* on page 2-18
ITATBCTR1	0xEF4	WO	-	See *ITATBCTR1 Register (ATB control 1)* on page 2-18
ITATBCTR0	0xEF8	WO	-	See *ITATBCTR0 Register (ATB control 0)* on page 2-19
Device Configuration	0xFC8	RO	0x00000000	See *Device Configuration Register* on page 2-9
Peripheral ID4	0xFD0	RO	0x00000004	See *Peripheral Identification Registers* on page 2-9
Peripheral ID5	0xFD4	RO	0x00000000	
Peripheral ID6	0xFD8	RO	0x00000000	
Peripheral ID7	0xFDC	RO	0x00000000	
Peripheral ID0	0xFE0	RO	0x0000000D	
Peripheral ID1	0xFE4	RO	0x000000B9	
Peripheral ID2	0xFE8	RO	0x000000XB[c]	
Peripheral ID3	0xFEC	RO	0x00000000	

Table 2-1 CoreSight ETM11 registers summary (continued)

Name	Base offset	Type	Reset value	Description
Component ID0	0xFF0	RO	0x000000B1	See *Component Identification Registers* on page 2-11
Component ID1	0xFF4	RO	0x00000090	
Component ID2	0xFF8	RO	0x00000005	
Component ID3	0xFFC	RO	0x000000B1	

a. Value given here is for the r1p1 release of the CoreSight ETM11. The reset value of bits[19:16] of the ID register depends on which ARM11 processor is implemented with the CoreSight ETM11. See *ETM ID Register* on page 2-5 for details.
b. The values of these read-only registers depend on the signals on external pins of the CoreSight ETM11. Therefore it is not possible to define the register reset values.
c. See *Peripheral Identification Registers* on page 2-9 for the value of X, bits [7:4] of the register value.

2.3 CoreSight ETM11 register descriptions

This section describes the following registers:

- *ETM ID Register*
- *Configuration Code Register* on page 2-7
- *Configuration Code Extension Register* on page 2-8
- *Device Configuration Register* on page 2-9
- *Peripheral Identification Registers* on page 2-9
- *Component Identification Registers* on page 2-11
- *Integration test registers* on page 2-12.

2.3.1 ETM ID Register

The ETM ID Register, at offset `0x1E4`, is read-only. Figure 2-1 shows the register bit assignments.

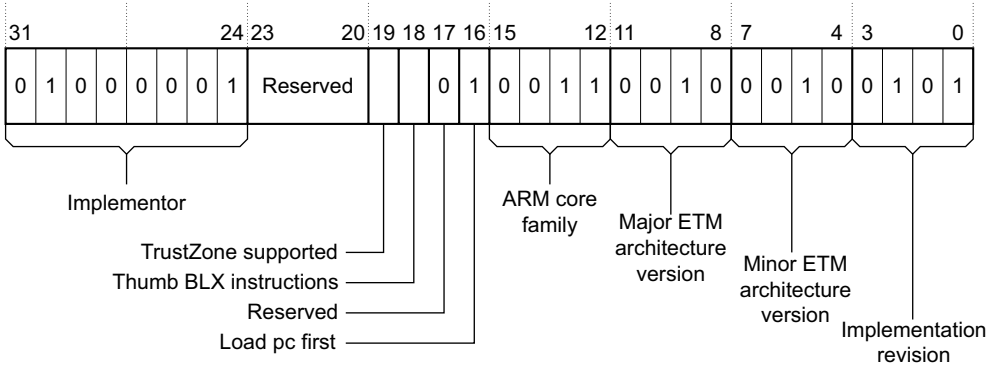


Figure 2-1 ETM ID Register bit assignments

Table 2-2 on page 2-6 describes the fields when reading the ID Register in the CoreSight ETM11. For the r1p1 release of CoreSight ETM11, the ETM ID Register has the following values:

- `0x41013225` when the CoreSight ETM11 is used with ARM1136J(F)-S
- `0x41053225` when the CoreSight ETM11 is used with ARM1156T2(F)-S
- `0x41093225` when the CoreSight ETM11 is used with ARM1176JZ(F)-S
- `0x41013225` when the CoreSight ETM11 is used with ARM11 MPCore.

Table 2-2 ETM ID Register bit assignments

Bit numbers	Value	Meaning
[31:24]	0x41	Implementor = A (ARM Limited).
[23:20]	b0000	Reserved. RAZ.
[19]	[a]	TrustZone support. There is no support for TrustZone on the ARM1136J(F)-S, ARM1156T2(F)-S and ARM11 MPCore processors. On CoreSight ETM11, this bit takes the value of the **TRUSTZONEEN** input pin. This bit must be set if the TrustZone architectural extensions are supported by the processor. • If this bit is set (1), some aspects of ETM behavior depend on whether the processor is in secure state or nonsecure state. • If this bit is not set (0), then the ETM behaves as if the processor is in secure state at all times.
[18]	[a]	Thumb-2 support. On CoreSight ETM11, this bit takes the value of the **THUMB2EN** input pin. This bit must be set if the Thumb-2 architectural extensions are supported by the processor. • If this bit is set (1), all 32-bit Thumb instructions are traced as a single instruction, including BL and BLX immediate. • If this bit is not set (0), BL and BLX immediate are traced as two instructions. Exceptions might occur between these instructions.
[17]	b0	Reserved.
[16]	b1	Load pc first. Special handling is not required to reconstruct the data addresses of an LDM with the pc in the register list because ETMv3.2 requires noncontiguous data addresses to be traced. However, special handling is required to determine which transfers correspond to which register.
[15:12]	b0011	ARM processor family = ARM11.
[11:8]	b0010	Major ETM architecture version number = 3.x
[7:4]	b0010	Minor ETM architecture version number = 2.
[3:0]	b0101	Implementation revision. The value given is for the r1p1 release of CoreSight ETM11.

a. Value depends on implementation. See *Meaning* column for details.

2.3.2 Configuration Code Register

The Configuration Code Register, at offset `0x004`, is read-only. Figure 2-2 shows the register bit assignments.

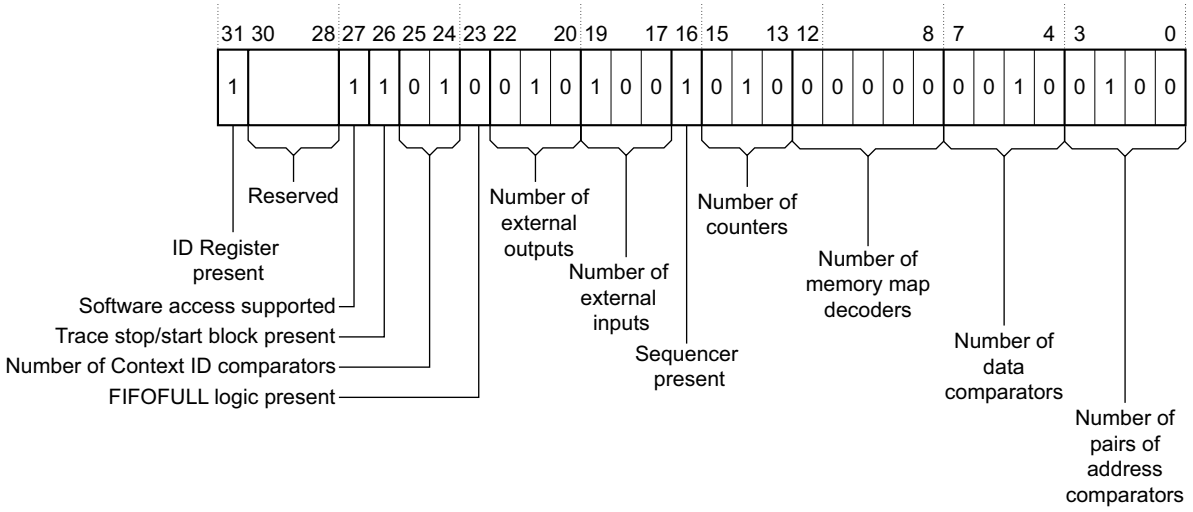


Figure 2-2 Configuration Code Register bit assignments

Table 2-3 lists the values of the fields when reading the Configuration Code Register. The Configuration Code Register has the value `0x8D294024`.

Table 2-3 Configuration Code Register bit assignments

Bits	Value	Meaning
[31]	1	CoreSight ETM11 ID Register present.
[30:28]	b000	Reserved. RAZ.
[27]	1	Software access is supported.
[26]	1	Trace start/stop block is present.
[25:24]	1	Number of Context ID comparators.
[23]	0	**FIFOFULL** logic absent.
[22:20]	2	Number of external outputs.
[19:17]	4	Number of external inputs.
[16]	1	The sequencer is present.

Table 2-3 Configuration Code Register bit assignments (continued)

Bits	Value	Meaning
[15:13]	2	Number of counters.
[12:8]	0	Number of memory map decoders.
[7:4]	2	Number of data comparators.
[3:0]	4	Number of pairs of address comparators.

2.3.3 Configuration Code Extension Register

The Configuration Code Extension Register, at offset `0x1E8`, is read-only. Figure 2-3 shows the register bit assignments.

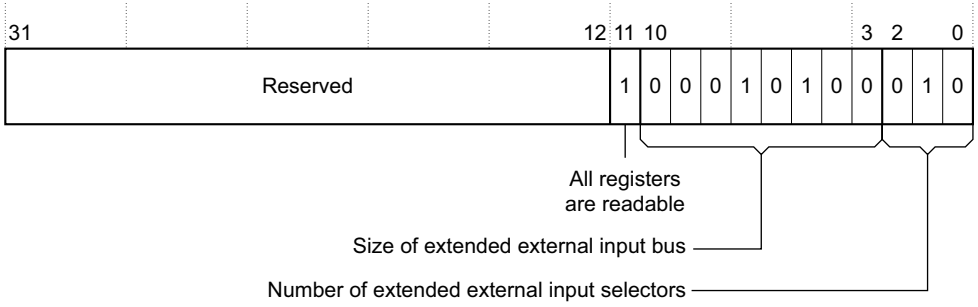


Figure 2-3 Configuration Code Extension Register bit assignments

Table 2-4 lists the value of the fields when reading the Configuration Code Extension Register. The Configuration Code Extension Register has the value `0x000008A2`.

Table 2-4 Configuration Code Extension Register bit assignments

Bit numbers	Value	Meaning
[31:12]	0	Reserved. RAZ.
[11]	1	All registers, except some integration test registers, are readable. See Table 2-1 on page 2-3 for details of the access allowed to integration test registers. (Registers with names which start with IT are the integration test registers, for example ITATBCTR1.)

Table 2-4 Configuration Code Extension Register bit assignments (continued)

Bit numbers	Value	Meaning
[10:3]	20	Size of extended external input bus.
[2:0]	2	Number of extended external input selectors.

2.3.4 Device Configuration Register

The Device Configuration Register, at offset `0xFC8`, is a 32-bit read-only register. This register defines CoreSight component capabilities. The Device Configuration Register always reads as `0x00000000`.

Note

There are no configuration options of the CoreSight ETM11 that are reflected in this register.

2.3.5 Peripheral Identification Registers

The CoreSight ETM11 Peripheral Identification Registers are a set of eight read-only registers, PeripheralID7 to PeripheralID0. These registers are defined in the *CoreSight Architecture Specification*, which specifies that only bits[7:0] are used. Table 2-5 lists the values of the fields when reading this set of registers. The *ETM Architecture Specification* gives a more detailed description of many of these fields.

Table 2-5 Peripheral Identification Registers, bit assignments

Register	Register offset	Bit	Value	Description
PeripheralID7	`0xFDC`	[31:8]	-	Unused, read undefined
		[7:0]	`0x00`	Reserved for future use, *Read-As-Zero* (RAZ)
PeripheralID6	`0xFD8`	[31:8]	-	Unused, read undefined
		[7:0]	`0x00`	Reserved for future use, RAZ
PeripheralID5	`0xFD4`	[31:8]	-	Unused, read undefined
		[7:0]	`0x00`	Reserved for future use, RAZ

Table 2-5 Peripheral Identification Registers, bit assignments (continued)

Register	Register offset	Bit	Value	Description
PeripheralID4	0xFD0	[31:8]	-	Unused, read undefined
		[7:4]	0x0	n, where 2^n is number of 4KB blocks used
		[3:0]	0x4	JEP106 continuation code [3:0]
PeripheralID3	0xFEC	[31:8]	-	Unused, read undefined
		[7:4]	0x0	RevAnd (at top level)
		[3:0]	0x0	Customer Modified 0x00 indicates from ARM
PeripheralID2	0xFE8	[31:8]	-	Unused, read undefined
		[7:4]	0x05	Revision Number of Peripheral. Value given is for r1p1 release.
		[3]	b1	Indicates that a JEDEC assigned value is used
		[2:0]	b011	JEP106 identity code [6:4]
PeripheralID1	0xFE4	[31:8]	-	Unused, read undefined
		[7:4]	0xB	JEP106 identity code [3:0]
		[3:0]	0x9	Part Number 1 Upper *Binary Coded Decimal* (BCD) value of Device Number
PeripheralID0	0xFE0	[31:8]	-	Unused, read undefined
		[7:0]	0x20	Part Number 0 Middle and Lower BCD value of Device Number

Note

In Table 2-5 on page 2-9, the *Peripheral Identification Registers* on page 2-9 are listed in order of register name, from most significant (ID7) to least significant (ID0). This does not match the order of the register offsets. Similarly, in Table 2-6 on page 2-11 the *Component Identification Registers* on page 2-11are listed in order of register name, from most significant (ID3) to least significant (ID0).

2.3.6 Component Identification Registers

There are four read-only Component Identification Registers, ComponentID3 to ComponentID0, that are defined in the *CoreSight Architecture Specification*. Although these are implemented as standard 32-bit registers:

- the most significant 24 bits of each register are not used and Read-As-Zero
- the least significant 8 bits of each register together make up the *Component ID*.

This concept of a single 32-bit Component ID, obtained from the four Component Identification Registers, is shown in Figure 2-4:

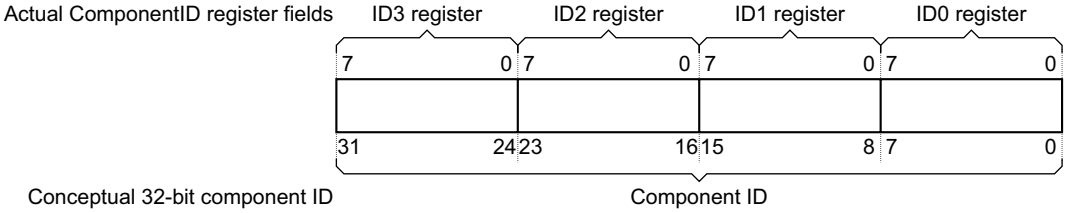


Figure 2-4 Mapping between the Component ID Registers and the Component ID value

Table 2-6 lists the values of the fields when reading the CoreSight ETM11 Component Identification Registers. This, again, shows how the valid fields combine to give the Component ID. This register structure is as defined in the *ETM Architecture Specification*.

Table 2-6 Component Identification Registers, bit assignments

Register	Register offset	Bit	Value	Description
ComponentID3	0xFFC	[31:8]	-	Unused, read undefined
		[7:0]	0xB1	Component identifier, bits[31:24]
ComponentID2	0xFF8	[31:8]	-	Unused, read undefined
		[7:0]	0x05	Component identifier, bits[23:16]
ComponentID1	0xFF4	[31:8]	-	Unused, read undefined
		[7:4]	0x9	Component class (component identifier, bits[15:12])
		[3:0]	0x0	Component identifier, bits[11:8]
ComponentID0	0xFF0	[31:8]	-	Unused, read undefined
		[7:0]	0x0D	Component identifier, bits[7:0]

2.3.7 Integration test registers

The following sub-sections describe the integration test registers. If you want to access these registers you must:

1. Set bit[0] of the Integration Mode Control Register to b1.
2. Clear bit[0] of the ETM Control Register to b0. If the read-only integration test registers are read while bit[0] is set, then an Unpredictable value might be returned.

You can use the write-only integration test registers to set the outputs of some of the ETM signals. Table 2-7 lists the signals which can be controlled in this way.

You can use the read-only integration test registers to read the state of some of the ETM input signals. Table 2-8 on page 2-13 lists the signals which can be read in this way.

See the *ETM Architecture Specification* for more information.

Table 2-7 Output signals that can be controlled by the integration test registers

Signal	Register	Bit	Register description
AFREADYM[a]	ITATBCTR0	[1]	See *ITATBCTR0 Register (ATB control 0)* on page 2-19
ATBYTESM[1:0][a]	ITATBCTR0	[9:8]	See *ITATBCTR0 Register (ATB control 0)* on page 2-19
ATDATAM[31, 23, 15, 7, 0][a]	ITATBDATA0	[4:0]	See *ITATBDATA0 Register (ATB data 0)* on page 2-17
ATIDM[6:0][a]	ITATBCTR1	[6:0]	See *ITATBCTR1 Register (ATB control 1)* on page 2-18
ATVALIDM[a]	ITATBCTR0	[0]	See *ITATBCTR0 Register (ATB control 0)* on page 2-19
EXTINACK[3:0]	ITMISCOUT	[3:0]	See *ITMISCOUT Register (miscellaneous outputs)* on page 2-14
EXTOUT[1:0]	ITMISCOUT	[9:8]	See *ITMISCOUT Register (miscellaneous outputs)* on page 2-14
TRIGOUT[a]	ITTRIGGERREQ	[0]	See *ITTRIGGERREQ Register (trigger request)* on page 2-16

a. These signals are only available with ETM11CS. Other signals are available with both ETM11CSSingle and ETM11CS.

Table 2-8 Input signals that can be read by the integration test registers

Signal	Register	Bit	Register description
AFVALIDM[a]	ITATBCTR2	[1]	See *ITATBCTR2 Register (ATB control 2)* on page 2-18
ATREADYM[a]	ITATBCTR2	[0]	See *ITATBCTR2 Register (ATB control 2)* on page 2-18
DBGACK	ITMISCIN	[4]	See *ITMISCIN Register (miscellaneous inputs)* on page 2-15
EXTIN[3:0]	ITMISCIN	[3:0]	See *ITMISCIN Register (miscellaneous inputs)* on page 2-15
EXTOUTACK[1:0]	ITMISCIN	[9:8]	See *ITMISCIN Register (miscellaneous inputs)* on page 2-15
TRIGOUTACK[a]	ITTRIGGERACK	[0]	See *ITTRIGGERACK Register (trigger acknowledge)* on page 2-16

a. These signals are only available with ETM11CS. Other signals are available with both ETM11CSSingle and ETM11CS

Using the integration test registers

The *CoreSight ETM11 Integration Manual* gives a full description of the use of the integration test registers to check integration. In brief:

- When bit 1 of the Integration Mode Control Register is set, values written to the write-only integration test registers map onto the specified outputs of CoreSight ETM11. For example, writing `0x3` to ITMISCOUT[9:8] causes **EXTOUT[1:0]** to take the value `0x3`.
- When bit 1 of the Integration Mode Control Register is set, values read from the read-only integration test registers correspond to the values of the specified inputs of CoreSight ETM11. For example, if you read ITMISCIN[9:8] you obtain the value of **EXTOUTACK[1:0]**.

ITMISCOUT Register (miscellaneous outputs)

The ITMISCOUT Register, at offset `0xEDC`, is write-only. Figure 2-5 shows the register bit assignments.

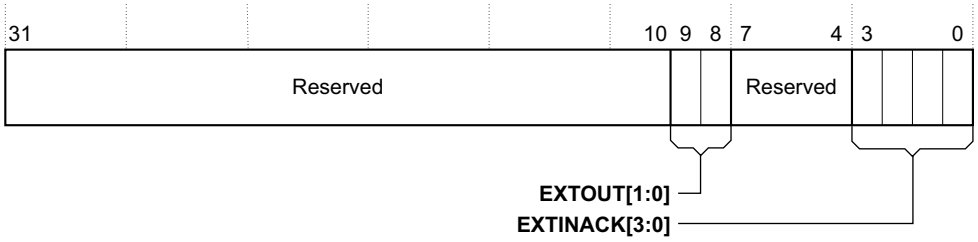


Figure 2-5 ITMISCOUT Register bit assignments

Table 2-9 lists the register bit assignments for the ITMISCOUT Register.

Table 2-9 ITMISCOUT Register bit assignments

Bits	Name	Function
[31:10]	-	Reserved. Write as zero.
[9:8]	EXTOUT	Drives the **EXTOUT[1:0]** output pins.
[7:4]	-	Reserved. Write as zero.
[3:0]	EXTINACK	Drives the **EXTINACK[3:0]** output pins.

ITMISCIN Register (miscellaneous inputs)

The ITMISCIN Register. at offset `0xEE0`, is read-only. Figure 2-6 shows the register bit assignments.

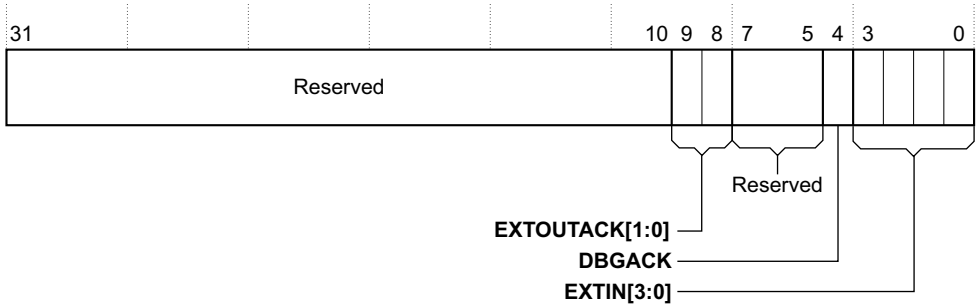


Figure 2-6 ITMISCIN Register bit assignments

Table 2-10 lists the fields when reading the ITMISCIN Register. The value of these fields depend on the signals on the input pins when the register is read.

Table 2-10 ITMISCIN Register bit assignments

Bits	Name	Function
[31:10]	-	Reserved. Read undefined.
[9:8]	EXTOUTACT	Returns the value of the **EXTOUTACK[1:0]** input pins.
[7:5]	-	Reserved. Read undefined.
[4]	DBGACK	Returns the value of the **DBGACK** input pins.
[3:0]	EXTIN	Returns the value of the **EXTIN[3:0]** input pins.

ITTRIGGERACK Register (trigger acknowledge)

The ITTRIGGERACK Register, at offset `0xEE4`, is read-only. Figure 2-7 shows the register bit assignments.

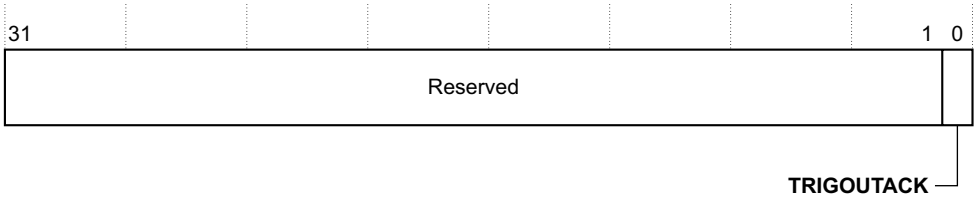


Figure 2-7 ITTRIGGERACK Register bit assignments

Table 2-11 describes the field when reading the ITTRIGGERACK Register. The value of this field depend on the signal on the input pins when the register is read.

Table 2-11 ITTRIGGERACK Register bit assignments

Bits	Name	Function
[31:1]	-	Reserved. Read undefined.
[0]	TRIGOUTACK	Returns the value of the **TRIGOUTACK** input pin.

ITTRIGGERREQ Register (trigger request)

The ITTRIGGERREQ Register, at offset `0xEE8`, is write-only. Figure 2-8 shows the register bit assignments.

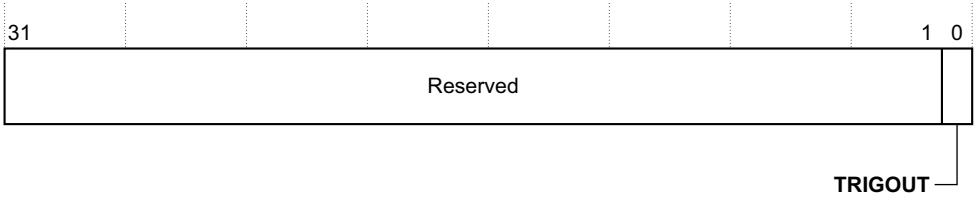


Figure 2-8 ITTRIGGERREQ Register bit assignments

Table 2-12 lists the register bit assignments for the ITTRIGGERREQ Register.

Table 2-12 ITTRIGGERREQ Register bit assignments

Bits	Name	Function
[31:1]	-	Reserved. Write as zero.
[0]	TRIGOUT	Drives the **TRIGOUT** output pin.

ITATBDATA0 Register (ATB data 0)

The ITATBDATA0 Register, at offset `0xEEC`, is write-only. Figure 2-9 shows the register bit assignments.

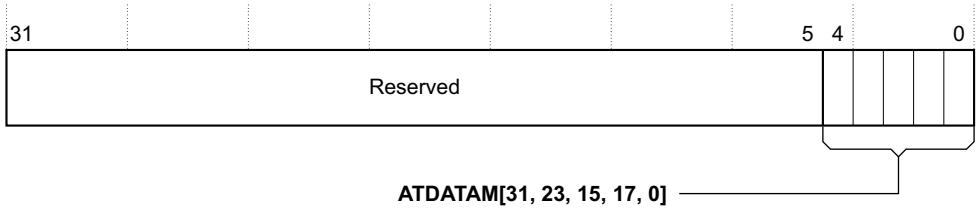


Figure 2-9 ITATBDATA0 Register bit assignments

Table 2-13 lists the register bit assignments for the ITATBDATA0 Register.

Table 2-13 ITATBDATA0 Register bit assignments

Bits	Name	Function
[31:5]	-	Reserved. Write as zero.
[4:0]	ATDATAM	Drives the **ATDATAM[31, 23, 15, 7, 0]** output pins.

ITATBCTR2 Register (ATB control 2)

The ITATBCTR2 Register, at offset `0xEF0`, is read-only. Figure 2-10 shows the register bit assignments.

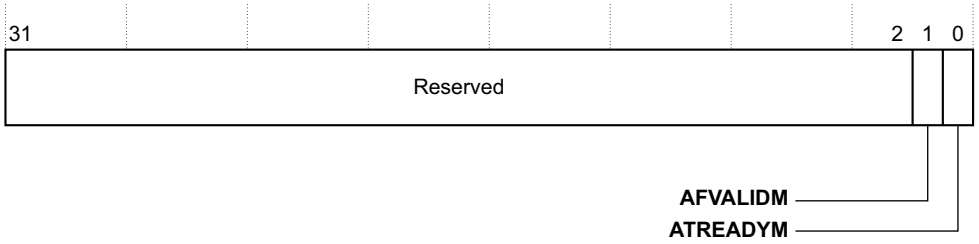


Figure 2-10 ITATBCTR2 Register bit assignments

Table 2-14 lists the fields when reading the ITATBCTR2 Register. The value of these fields depend on the signals on the input pins when the register is read.

Table 2-14 ITATBCTR2 Register bit assignments

Bits	Name	Function
[31:2]	-	Reserved. Read undefined.
[1]	AFVALIDM	Returns the value of the **AFVALIDM** input pin.
[0]	ATREADYM	Returns the value of the **ATREADYM** input pin.

ITATBCTR1 Register (ATB control 1)

The ITATBCTR1 Register, at offset `0xEF4`, is write-only. Figure 2-11 shows the register bit assignments.

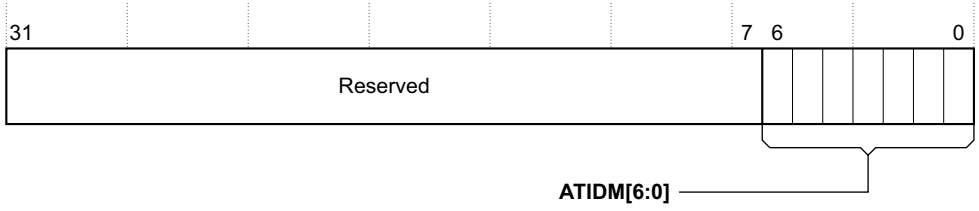


Figure 2-11 ITATBCTR1 Register bit assignments

Table 2-15 lists the register bit assignments for the ITATBCTR1 Register.

Table 2-15 ITATBCTR1 Register bit assignments

Bits	Name	Function
[31:7]	-	Reserved. Write as zero.
[6:0]	ATIDM	Drives the **ATIDM[6:0]** output pins.

ITATBCTR0 Register (ATB control 0)

The ITATBCTR0 Register, at offset `0xEF8`, is write-only. Figure 2-12 shows the register bit assignments.

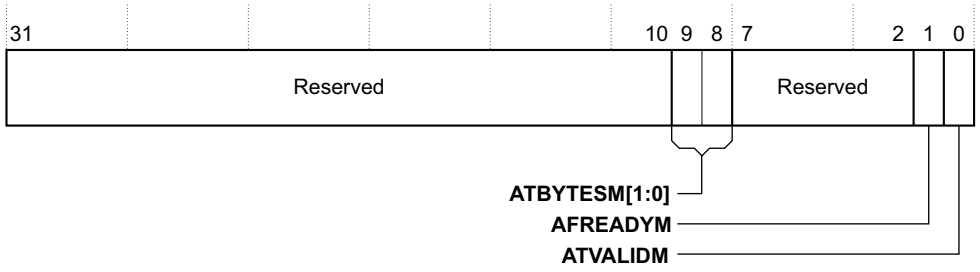


Figure 2-12 ITATBCTR0 Register bit assignments

Table 2-16 lists the register bit assignments for the ITATBCTR0 Register.

Table 2-16 ITATBCTR0 Register bit assignments

Bits	Name	Function
[31:10]	-	Reserved. Write as zero.
[9:8]	ATBYTESM	Drives the **ATBYTESM[1:0]** output pins.
[7:2]	-	Reserved. Write as zero.
[1]	AFREADYM	Drives the **AFREADYM** output pin.
[0]	ATVALIDM	Drives the **ATVALIDM** output pin.

2.4 Precise TraceEnable events

The *ETM Architecture Specification* states that **TraceEnable** is Imprecise under certain conditions, with some implementation-defined exceptions. In CoreSight ETM11 if the enabling event register selects the following resources it does not cause **TraceEnable** to be Imprecise, provided that the resources are themselves precise:

- single address comparators
- address range comparators.

For more information on Precise TraceEnable events, see the *ETM Architecture Specification*.

2.5 Parallel instruction execution

ARM11 processors support *branch folding*, where correctly predicted branches are executed in parallel with the following instruction. This means the CoreSight ETM11 is capable of tracing two instructions per cycle, although only the second instruction can have data associated with it.

Although the trace start/stop block is evaluated for each instruction as required, CoreSight ETM11 cannot trace one instruction of a folded branch without the other. In particular, if a folded branch is traced, the instruction it is paired with is also traced. If **ViewData** is active, any data associated with the paired instruction is also traced.

2.6 Support for independent load/store units

ARM11 processors have independent load/store units. This means that they are capable of continuing to transfer data values for an earlier instruction while later instructions are executing. In these circumstances, the later instructions are said to have executed underneath the data instruction. When a data instruction has been traced, all instructions executed underneath it are also traced.

Note

This does not create much additional trace, because the instructions executed underneath cannot be data instructions or indirect branches.

The behavior of the ARM11 load/store units enables data transfers to complete out of order. This means that the data for a data instruction can return later than the data for a subsequent data instruction. This results in CoreSight ETM11 generating a out-of-order placeholder and out-of-order data packets.

The out-of-order behavior of data transfers also affects the behavior of the address comparators, especially when used with the Exact Match bit (bit[7]) of the Address Access Type Register set. For more details on out-of-order data transfers associated with the comparator operation, see the *ETM Architecture Specification*.

When the Exact Match bit is set, the address range comparators only match on the cycle that the instruction or data transfer occurs. The comparators do not hold their value on interim cycles.

Note

Using data values to create an event, such as a sequencer transition, might result in out-of-order events occurring because the load data can be returned out of order. If you are concerned about this behavior, you can disable *Hit-Under-Miss* (HUM) functionality in the core by using the low interrupt latency configuration, that is, writing to the FI bit (bit[21]) of the cp15 Control Register (c1). See the relevant ARM11 TRM for more information.

2.7 Context ID tracing

CoreSight ETM11 detects the MCR instruction that changes the context ID, and generates the appropriate number of bytes as a context ID packet instead of a normal data packet. As a result, if context ID tracing is enabled, an MCR instruction that changes the context ID does not have its data traced separately.

Because the ARM1176JZ(F)-S has a secure and a nonsecure context ID, the ETM11 outputs a context ID when switching occurs between secure and nonsecure states.

2.8 Interaction with the Performance Monitoring Unit, PMU

Note

This section is not applicable for ARM11 MPCore processors.

ARM11x6 processors include a PMU that enables events, such as cache misses and instructions executed, to be counted over a period of time. These events are all available for use by CoreSight ETM11 using the extended external input facility. Each bit in the **EVNTBUS[19:0]** input is mapped to the corresponding extended external input. See the relevant ARM11x6 TRM for details of the mapping of events to bits within this bus.

Some events use two bits. Two of these events can occur in a cycle. They must be dealt with separately if they are to be properly counted.

ARM11x6 PMUs can use the two CoreSight ETM11 external outputs as additional events. These events are not provided back to CoreSight ETM11 as extended external inputs.

These facilities enable the system events to be further qualified with ETM resources, such as instruction address ranges or the start/stop resource, before being passed back to the PMU for counting. This can be done as follows:

- Configure the ETM extended external input selectors to the system events you want to count.
- Configure the desired ETM filtering resource as appropriate.
- Configure the ETM external outputs to extended external input selector AND the desired ETM filtering resource.
- Select the ETM external outputs as the events to be counted in the ARM11x6 PMU.

2.9 ETM11CSSingle clocks

This section describes ETM11CSSingle clocks and clock enables:

- *ETM11CSSingle clock signals*
- *ETM11CSSingle clock enable signals* on page 2-26.

2.9.1 ETM11CSSingle clock signals

ETM11CSSingle contains the following clocks:

CLK This is the main clock for the ETM11CSSingle block and must be the same clock as that wired to the **CLK** input of the ARM11 processor. It clocks all registers in the core-facing logic of ETM11CSSingle. It can be asynchronous to **PCLK and ATCLK**.

PCLK This is the system APB interface clock in ETM11CSSingle. It can be asynchronous to **CLK** and **ATCLK**.

ATCLK This is the *AMBA Trace Bus* (ATB) interface clock. It can be asynchronous to **CLK** and **PCLK**.

Note

The effective **PCLK** frequency must be the same or slower than the effective **ATCLK** frequency:

- the effective **PCLK** frequency is obtained from **PCLK** and **PCLKEN**
- the effective **ATCLK** frequency is obtained from **ATCLK** and **ATCLKEN**.

TRACECLK Trace clock generated in ETMTRACEPORT for use off-chip.

TRACECLK is generated from **ATCLK**. It is exported off-chip for the trace capture device to clock in the trace data. It is not used anywhere else in ETM11CSSingle. During layout, the rising edge of **TRACECLK** must be delayed to the middle of the **ATCLK** cycle to provide maximum set-up time for trace capture devices to sample the trace data off **TRACECLK**. See the *CoreSight ETM11 Configuration and Sign-off Guide* for details.

2.9.2 ETM11CSSingle clock enable signals

ETM11CSSingle has the following clock enable signals:

PCLKEN This is the system APB interface clock enable. It can used to interface to a slower **PCLK** domain.

ATCLKEN This is the ATB interface clock enable. It can be used to interface to a slower **ATCLK** domain.

2.10 ETM11CSSingle resets

ETM11CSSingle contains the following resets:

nPORESET This signal is the main power-on reset. It resets registers in the CLK domain of ETM11CSSingle. It is active LOW.

PRESETn This signal is the system APB reset. It resets registers in the PCLK domain of ETM11CSSingle except the TAP controller registers in ETMJTAGPORT. It is active LOW.

ATRESETn This signal is the ATB interface reset. It resets registers in the ATCLK domain of ETM11CSSingle. It is active LOW.

nTRST This signal is the JTAG reset for ETM11CSSingle. It resets registers in the TAP controller of ETMJTAGPORT. It is active LOW.

2.11 ETM11CS clocks

This section describes ETM11CS clocks and clock enables:

- *ETM11CS clock signals*
- *ETM11CS clock enable signals*.

2.11.1 ETM11CS clock signals

ETM11CS contains the following clocks:

CLK This is the main clock for the ETM11CS block and must be the same clock as that wired to the **CLK** input of the ARM11 processor. It can be asynchronous to **PCLKDBG** and **ATCLK**.

PCLKDBG This is the Debug APB interface clock for ETM11CS. It can be asynchronous to **CLK** and **ATCLK**.

ATCLK This is the ATB interface clock. It can be asynchronous to **CLK** and **PCLKDBG**.

2.11.2 ETM11CS clock enable signals

ETM11CS has the following clock enable signals:

ATCLKEN This is the ATB clock enable. It can be used to interface to a slower **ATCLK** domain.

PCLKENDBG This is the Debug APB interface clock enable. It can be used to interface to a slower **PCLKDBG** domain.

2.12 ETM11CS resets

ETM11CS contains the following resets:

nPORESET This signal is the main power-on reset. It resets all the registers in the CLK domain of ETM11CS. It is active LOW.

PRESETDBGn This signal is the Debug APB interface reset. It resets all the registers in the PCLKDBG domain of ETM11CS. It is active LOW.

ATRESETn This signal is the ATB interface reset. It resets all the registers in the ATCLK domain. It is active LOW.

2.13 PORTMODE, PORTSIZE and MAXPORTSIZE

PORTMODE and **PORTSIZE** can be set by writing to the ETM Control Register, `0x00`. See *ETM Architecture Specification* for more details.

2.13.1 PORTMODE and PORTSIZE in ETM11CS

In a CoreSight system **PORTMODE** and **PORTSIZE** are not used.

2.13.2 PORTMODE and PORTSIZE in ETM11CSSingle

In ETM11CSSingle, **PORTMODE** and **PORTSIZE** are used to control ETMTRACEPORT functionality.

ETM11CSSingle supports the following port modes and given combinations of **PORTMODE** and **PORTSIZE**:

Dynamic mode, PORTMODE[2:0] = b000

ETMTRACEPORT functionality is disabled. **PORTSIZE** must be set to 32 bits.

Asynchronous mode, PORTMODE[2:0] = b100

ETMTRACEPORT functionality is enabled.

Table 2-17 lists the supported **PORTMODE** and **PORTSIZE** combinations.

Table 2-17 Supported PORTMODE and PORTSIZE combinations

PORTMODE[2:0] value	PORTSIZE[3:0] value	Mode
b000	b0100 (32-bit port)	Dynamic
b100	b0100 (32-bit port)	Asynchronous
b100	b0010 (16-bit port)	Asynchronous
b100	b0001 (8-bit port)	Asynchronous
b100	b0000 (4-bit port)	Asynchronous
b100	b1001 (2-bit port)	Asynchronous

2.13.3 MAXPORTSIZE in ETM11CS and ETM11CSSingle

MAXPORTSIZE can be set to any architecturally defined value.

MAXPORTSIZE appears in the System Configuration Register. If CoreSight ETM11 is programmed with a port size greater than that specified by **MAXPORTSIZE**, the port size supported bit, bit 10 of the ETM System Configuration Register, is LOW. The debugger must check that this bit is HIGH before a debugging session to ensure that a valid port size has been programmed.

2.14 Data instructions in Java state

Table 2-18 gives the amount of trace expected for each data bytecode. This is Implementation Defined but common to ETM10RV, ETM11RV and CoreSight ETM11.

Table 2-18 Bytecodes and amounts of trace

Bytecodes	Amount of trace
baload, bastore	Byte
caload, castore, saload, sastore	Halfword
iload, aload, fload, iaload, aaload, faload, istore, astore, fstore, iastore, aastore, fastore, net, getfield_a, putfield_a, ldc_a, ldc_w_a, getstatic_a (SP=1), putstatic_a (SP=1)	Word
dload, lload, daload, laload, dstore, lstore, dastore, lastore getfield2_a, putfield2_a, ldc2_w_a, getstatic_a (SP=0), putstatic_a (SP=0)	Two words
all other bytecodes	no data

Table 2-18 assumes the bytecode is implemented in hardware. The Jazelle specification allows any bytecode to be implemented in software, and normal ARM state data trace is output instead in this case. This applies to aastore in current Jazelle implementations.

2.15 Restrictions and limitations

See your ARM11 processor TRM for any restrictions or limitations on what can be traced.

Chapter 3
Programmer's Model

This chapter describes the mechanisms for programming the registers used to set up the trace and triggering facilities of CoreSight ETM11. It contains the following sections:

3.1 About the programmer's model

When programming the ETM registers you must enable all the changes at the same time. For example, if the counter is reprogrammed, it might start to count based on incorrect events, before the trigger condition has been correctly set up.

You can use the ETM programming bit in the ETM Control Register to disable all trace operations during programming. To do this follow the procedure shown in Figure 3-1 on page 3-3.

The individual registers are not described here. For more information, see the *ETM Architecture Specification*.

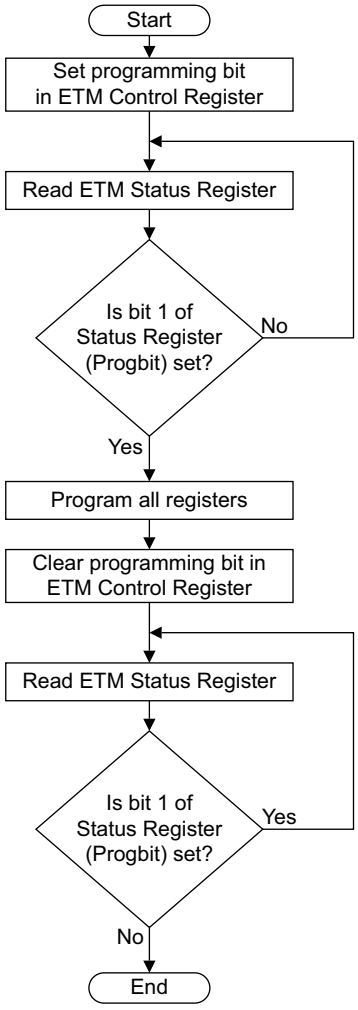


Figure 3-1 Programming ETM registers

The processor does not need to be in the debug state while the ETM registers are being programmed.

3.2 Programming and reading ETM registers

There are two methods of access to the CoreSight ETM11 registers:

- *Software access using APB*
- *JTAG access through ETMJTAGPORT (JTAG-APB bridge) in ETM11CSSingle*.

3.2.1 Software access using APB

APB provides a direct method of programming stand-alone CoreSight ETM11 (ETM11CSSingle) and CoreSight ETM11 in a CoreSight system.

APB is a simple low cost interface used to provide access to the programmable control registers of peripheral devices. It has the following features:

- Unpipelined protocol, that is, a second transfer cannot start before the first transfer completes.
- Every transfer takes at least two cycles.

For more information on APB transfers, see *AMBA 3 APB Protocol Specification.*

3.2.2 JTAG access through ETMJTAGPORT (JTAG-APB bridge) in ETM11CSSingle

JTAG programming is through the ETMJTAGPORT described in *ETMJTAGPORT* on page 4-3. The interface is an extension of the ARM TAP controller, and is assigned scan chain number 6. The scan chain consists of a 40-bit shift register comprising:

- A 32-bit data field
- A 7-bit address field
- A read/write bit.

Some registers in CoreSight ETM11 are outside the address range of the JTAG interface. These include the CoreSight Management Registers and the CoreSight ETM11 Integration Registers, which are accessible via APB. The CoreSight Management Registers are only used in a CoreSight system. The general arrangement of the ETM JTAG registers is shown in Figure 3-2 on page 3-5.

When writing a register, the data to be written is scanned into the 32-bit data field, the address of the register into the 7-bit address field and a 1 into the read/write bit. A register is read by scanning its address into the address field and a 0 into the read/write bit. The 32-bit data field is ignored.

The read data is transferred into the lower 32-bit data field when the TAP state machine transitions through CAPTURE-DR.

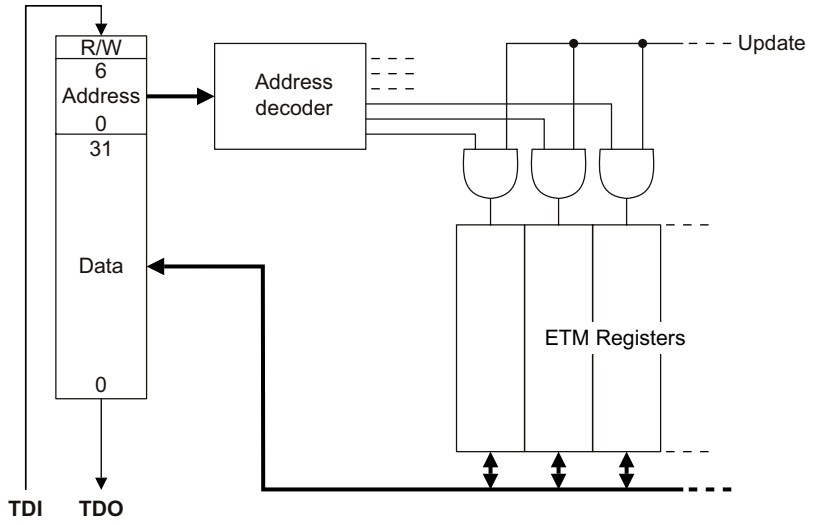


Figure 3-2 ETM JTAG structure

Note

A read or write takes place when the TAP controller enters the UPDATE-DR state.

Chapter 4
Blocks for Stand-alone CoreSight ETM11

This chapter contains information about the additional blocks for a stand-alone CoreSight ETM11. It contains the following section:

4.1 About additional blocks for stand-alone CoreSight ETM11

Two additional components are available for stand-alone CoreSight ETM11:

ETMJTAGPORT

A JTAG to APB bridge that supports the JTAG programming model for access to the ETM11.

ETMTRACEPORT

A Trace Port Interface Unit for tracing a single trace source that supports multiple trace port sizes.

Figure 4-1 shows CoreSight ETM11 with an ETMJTAGPORT and ETMTRACEPORT.

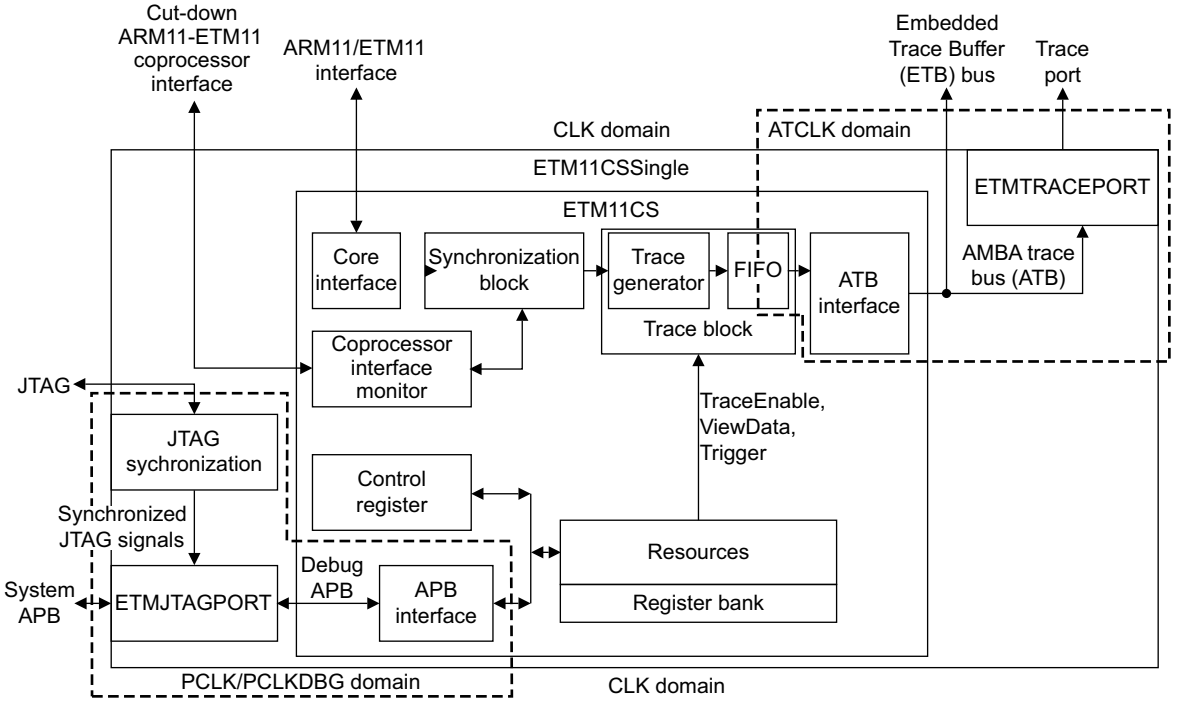


Figure 4-1 CoreSight ETM11 with ETMJTAGPORT and ETMTRACEPORT

4.1.1 ETMJTAGPORT

ETMJTAGPORT converts synchronized JTAG programming commands to APB accesses. ETMJTAGPORT arbitrates between synchronized JTAG and system APB. If a JTAG access is in progress, the system **PREADY** is held LOW for the duration of the JTAG access. This is because APB access can be stalled but JTAG access cannot be stalled.

ETMJTAGPORT contains a TAP controller that must be placed in parallel with the core TAP controller when ETMJTAGPORT is connected to a CoreSight ETM11, so that both TAP controllers are in step with one another. Figure 4-2 shows an example of TAP controllers in step and an optional ETB11.

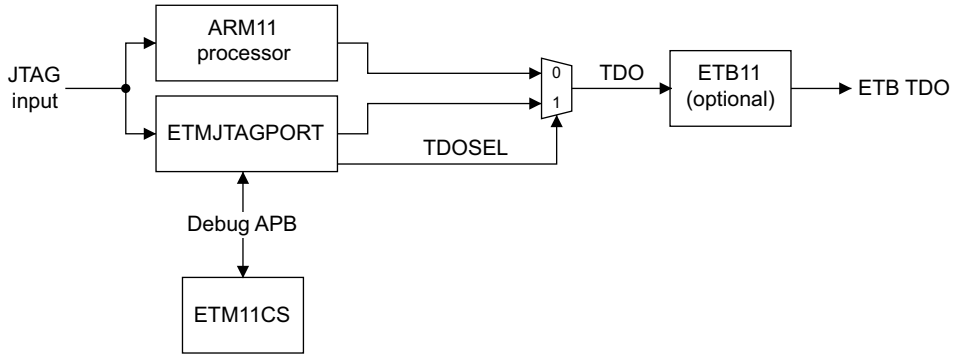


Figure 4-2 TAP controllers in step

4.1.2 JTAG synchronization

The registers in ETMJTAGPORT are clocked by the effective **PCLK** (**PCLK** + **PCLKEN**). Therefore its JTAG interface inputs must be synchronized to the effective **PCLK**. This is done in the JTAG synchronization module, ETMJTAGSync, of ETM11CSSingle as shown in Figure 4-3 on page 4-4.

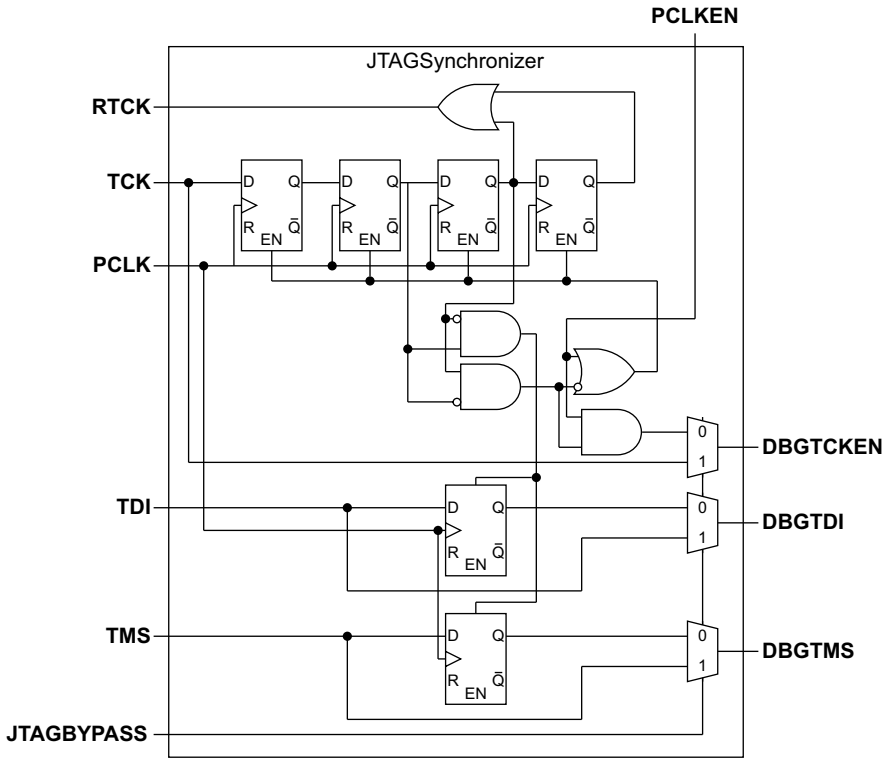


Figure 4-3 JTAG synchronization block

Note

All registers in Figure 4-3 are reset by **nTRST**.

To set **JTAGSBYPASS** to 0:

- The effective **PCLK** must be the same or slower than the effective **ATCLK**.

To set **JTAGSBYPASS** to 1:

- **PCLK** must be identical to **CLK**.
- The effective **PCLK** must be the same or slower than the effective **ATCLK**.
- Either:
  - **PCLKEN** must be set to 1
  - **TCKEN** from the external JTAG synchronizer must be a pure subset of **PCLKEN**.

Note

See the *CoreSight ETM11 Integration Manual* for details of JTAG scan chain connections.

4.1.3 Programmer's model for stand-alone systems

This section describes:

- *The Scan Chain 6 Register*.

The Scan Chain 6 Register

To write an ETM register, bit[39] must be set and the address field, bits[38:32], must contain a valid ETM register address. The data in bits[31:0] are then written into the corresponding ETM register when the TAP controller goes through the UpdateDR state.

To read an ETM register, bit[39] must be clear and the address field must contain a valid ETM register address. The register data is fetched from the ETM11 when the TAP controller passes through the UpdateDR state, and loaded into the Scan Chain 6 Register when the TAP controller passes through the CaptureDR state. This data can then be read out through **TDO** by putting the TAP controller into the ShiftDR state.

Table 4-1 describes the format of the 40-bit Scan Chain 6 Register

Table 4-1 Scan Chain 6 Register bit assignments

Bit	Description
[39]	Write /!Read Access
[38:32]	Register Address
[31:0]	Data

4.1.4 ETMTRACEPORT

ETMTRACEPORT takes the ETM11 trace output stream and converts it to a form suitable for taking off-chip. It also creates and exports a clock, **TRACECLK**, used by the off-chip capture device to sample the converted trace data.

TRACECLK and its associated data are generated from **ATCLK**. CoreSight ETM11CSSingle supports asynchronous CLK, PCLK, and ATCLK clock domains. Therefore, it is not necessary for the trace port clock frequency to be an integer division of the main clock.

ETMTRACEPORT sets its trace port configuration according to the values of **PORTSIZE[3:0]** output from CoreSight ETM11. ETMTRACEPORT only supports trace port sizes of 2, 4, 8, 16 and 32 bits, because ETMTRACEPORT only receives 32-bit data packets from the ETM11, and all these port sizes wholly divide into 32.

See the *CoreSight ETM11 Configuration and Sign-off Guide* for implementation details about delaying **TRACECLK**.

Appendix A
Signals Lists

This appendix describes the signals used in the CoreSight ETM11. It contains the following sections:

A.1 ETM11CSSingle Signals

Table A-1 lists the ETM11CSSingle signals in alphabetical order. Clock domains, where specified, give the clock on which input signals must be generated and output signals sampled. See the *CoreSight ETM11 Integration Manual* for information about signals and connectivity.

Table A-1 ETM11CSSingle signals

Signal name	Input/ output	Description	Domains	
			Clock	Power
ASICCTL[7:0]	Output	Contents of the ASIC control register.	**CLK**	V_{CORE}
ATCLK	Input	ATB interface clock.	-	V_{SOC}
ATCLKEN	Input	Enable signal for **ATCLK.**	**ATCLK**	V_{SOC}
ATRESETn	Input	Internal trace bus reset. Resets all registers in the ATCLK domain. Active LOW.	Internally synchronized	V_{SOC}
CFGBIGEND	Input	Output signal from ARM11 family cores. **CFGBIGEND** at logic 1 indicates that the core is operating in big-endian (BE-32) mode.	**CLK**	V_{SOC}
CLK	Input	The main clock for the ETM11CSSingle.	-	V_{SOC}
CORESELECT[2:0]	Output	If the ETM is shared between multiple cores, this signal specifies which core to trace.	-	V_{CORE}
DBGACK	Input	Indicates that the core is in debug state. This signal is connected to the core general purpose **DBGACK** output, so that it can be used to determine when **ETMDBGRQ** can be deasserted. It is also used for other purposes in the ETM, and care must be taken to ensure the timing of this signal is appropriate because it does not come through the main core/ETM interface.	**CLK**	V_{SOC}
DBGEN	Input	Invasive debug enable. At logic 1 indicates that invasive debug is enabled.	-	V_{SOC}
ETBTRACEDATA[31:0]	Output	Provides trace data for ETB capture.	**ATCLK**	V_{SOC}
ETBTRACEVALID	Output	Indicates that trace data is valid.	**ATCLK**	V_{SOC}
ETBTRIGGER	Output	Indicates trigger condition.	**ATCLK**	V_{SOC}

Table A-1 ETM11CSSingle signals (continued)

Signal name	Input/ output	Description	Domains	
			Clock	Power
ETMCPADDRESS[14:0]	Input	Coprocessor interface address. The relevant fields of the MRC or MCR instruction are encoded here as follows: Bits 14:12 <Opcode_1> Bits 11:8 <CRn> Bits 7:4 <CRm> Bit 3 = 0 cp14 Bit 3 = 1 cp15 Bits 2:0 <Opcode_2>.	**CLK**	V_{CORE}
ETMCPCOMMIT	Input	Coprocessor interface commit If this is LOW two cycles after **ETMCPENABLE** is asserted, the transfer is canceled and must not take any effect.	**CLK**	V_{CORE}
ETMCPENABLE	Input	Coprocessor interface enable **ETMCPWRITE** and **ETMCPADDRESS** are valid this cycle, and the remaining signals are valid two cycles later.	**CLK**	V_{CORE}
ETMCPSECCTL[1:0]	Input	Coprocessor interface security control	**CLK**	V_{CORE}
ETMCPWDATA[31:0]	Input	Coprocessor interface write value.	**CLK**	V_{CORE}
ETMCPWRITE	Input	Coprocessor interface read or write. This signal is asserted for writes.	**CLK**	V_{CORE}
ETMDA[31:3]	Input	Address for data transfer.	**CLK**	V_{CORE}
ETMDACTL[17:0]	Input	Data address interface control signals.	**CLK**	V_{CORE}
ETMDBGRQ	Output	Request from the CoreSight ETM11 for the core to enter debug state. This must be ORed with any ASIC-level **DBGRQ** signals before being connected to the core **EDBGRQ** input.	**CLK**	V_{CORE}
ETMDD[63:0]	Input	Contains the data value for a Load, Store, MRC, or MCR instruction.	**CLK**	V_{CORE}
ETMDDCTL[3:0]	Input	Data interface control signals.	**CLK**	V_{CORE}

Table A-1 ETM11CSSingle signals (continued)

Signal name	Input/ output	Description	Domains	
			Clock	Power
ETMDD2[63:0]	Input	Contains the data value for a Store or MCR instruction. Only for use with an ARM11 MPCore processor.	**CLK**	V_{CORE}
ETMDDCTL2[1:0]	Input	Data interface control signals. Only for use with an ARM11 MPCore processor.	**CLK**	V_{CORE}
ETMEN	Output	When HIGH, ETMTRACEPORT is enabled. When LOW, logic related to the to the ETMTRACEPORT can be clock-gated.	**CLK**	V_{CORE}
ETMIA[31:0]	Input	Address for executed instruction.	**CLK**	V_{CORE}
ETMIACTL[17:0]	Input	Instruction address interface control signals.	**CLK**	V_{CORE}
ETMIARET[31:0]	Input	Address to return to if branch is incorrectly predicted.	**CLK**	V_{CORE}
ETMIASECCTL[1:0]	Input	Instruction interface security control signals	**CLK**	V_{CORE}
ETMPADV[2:0]	Input	Pipeline advance signals.	**CLK**	V_{CORE}
ETMPWRUP	Output	When HIGH, indicates that CoreSight ETM11 is in use. When LOW: • external logic supporting CoreSight ETM11 can be clock-gated to conserve power. • the ARM11 processor disables the CoreSight ETM11 interface • logic within CoreSight ETM11 is clock-gated to conserve power.	**CLK**	V_{CORE}
ETMWFIPENDING	Input	Indicates that the ARM11 processor is about to go into Standby mode, and that the ETM must drain its FIFO.	**CLK**	V_{CORE}
EVNTBUS[19:0]	Input	Gives the status of the processor performance monitoring events. Available as extended external inputs events. Not used on ARM11 MPCore processor.	**CLK**	V_{CORE}
EXTIN[3:0]	Input	External input resources.	See footnote [a]	V_{CORE}

Table A-1 ETM11CSSingle signals (continued)

Signal name	Input/ output	Description	Domains	
			Clock	Power
EXTINACK[3:0]	Output	Acknowledge signal for the **EXTIN[3:0]** bus. When **EXTSBYPASS** is HIGH, **EXTINACK[3:0]** are not valid and can be ignored.	-	V_{CORE}
EXTOUT[1:0]	Output	External outputs.	See footnote [a]	V_{CORE}
EXTOUTACK[1:0]	Input	Acknowledge signals for the **EXTOUT[1:0]** bus. When **EXTSBYPASS** is HIGH, these signals must be tied LOW.	-	V_{CORE}
EXTSBYPASS	Input	**EXTSBYPASS** is a single bit input to the ETM that is used to bypass the synchronization of the external inputs **EXTIN[3:0]** and external outputs **EXTOUT[1:0]**. When EXTSBYPASS is LOW: • a HIGH output on any bit of **EXTOUT[1:0]** is held until the corresponding bit of **EXTOUTACK[1:0]** is asserted HIGH • a HIGH on any bit of **EXTIN[3:0]** must remain held until the ETM asserts the corresponding bit of **EXTINACK[3:0]** HIGH. When **EXTSBYPASS** is HIGH, **EXTOUT[1:0]** outputs are valid on the rising edge of **CLK** and **EXTIN[3:0]** inputs must be valid on the rising edge of **CLK**.	-	V_{CORE}
FIFOPEEK[8:0]	Output	For validation purposes only. Indicates when various events occur before being written to the FIFO.	**CLK**	V_{CORE}
JTAGSBYPASS	Input	**JTAGSBYPASS** is a single bit input to the ETM that is used to bypass the JTAG synchronization.	-	V_{SOC}
MAXCORES[2:0]	Input	Where a CoreSight ETM11 is shared between multiple cores, this signal specifies the number of cores the ETM can trace. It must be tied to the number of cores sharing the ETM minus 1.	-	V_{CORE}
MAXEXTIN[2:0]	Input	External inputs supported by the ASIC (maximum 4). This appears in the configuration code register.	-	V_{CORE}

Table A-1 ETM11CSSingle signals (continued)

Signal name	Input/ output	Description	Domains	
			Clock	Power
MAXEXTOUT[1:0]	Input	External outputs supported by the ASIC (maximum 2). This appears in the configuration code register.	-	V_{CORE}
MAXPORTSIZE[3:0]	Input	Maximum port size supported. See *MAXPORTSIZE in ETM11CS and ETM11CSSingle* on page 2-30.	-	V_{CORE}
nCORECLAMP	Input	Indicates power down of the Core domain. Active LOW.	-	V_{SOC}
nETMWFIREADY	Output	Indicates that CoreSight ETM11 FIFO is empty and that the ARM11 processor can be put into Standby mode.	**CLK**	V_{CORE}
NIDEN	Input	Non-invasive debug enable.	Internally synchronized	V_{SOC}
nPORESET	Input	Main power on reset. Resets all registers in the CLK domain.	Internally synchronized	V_{SOC}
nSOCCLAMP	Input	Indicates power down of the SoC domain. Active LOW.	-	V_{SOC}
nTRST	Input	JTAG interface reset. Active LOW.	-	V_{SOC}
PADDR[11:2]	Input	System APB Address Bus.	**PCLK**	V_{SOC}
PCLK	Input	System APB interface clock	-	V_{SOC}
PCLKEN	Input	System APB clock enable.	**PCLK**	V_{SOC}
PENABLE	Input	The system APB interface is enabled for a transfer.	**PCLK**	V_{SOC}
PORTMODE[2:0]	Output	Currently requested port mode.	**CLK**	V_{SOC}
PORTSIZE[3:0]	Output	Currently requested port size.	**CLK**	V_{SOC}
PRDATA[31:0]	Output	System APB read data.	**PCLK**	V_{SOC}
PREADY	Output	Used to extend transfers, and to arbitrate between JTAG and APB accesses.	**PCLK**	V_{SOC}

Table A-1 ETM11CSSingle signals (continued)

Signal name	Input/ output	Description	Domains	
			Clock	Power
PRESETn	Input	System APB interface reset. Active LOW.	Internally synchronized	V_{SOC}
PSEL	Input	System APB select signal.	**PCLK**	V_{SOC}
PSLVERR	Output	System APB error signal	**PCLK**	V_{SOC}
PWDATA[31:0]	Input	System APB write data.	**PCLK**	V_{SOC}
PWRITE	Input	System APB transfer direction (!Read/Write).	**PCLK**	V_{SOC}
RSTBYPASS	Input	Reset synchronization bypass.	-	V_{SOC}
RTCK	Output	JTAG interface return clock	-	V_{SOC}
SE	Input	Scan enable.	-	V_{SOC}
TCK	Input	JTAG interface clock.	-	V_{SOC}
TDI	Input	Test data in. Must be connected to **TDI** input to the core.	See footnote [b]	V_{SOC}
TDO	Output	Test data out. Must be multiplexed externally with TDO from the core, controlled by **TDOSEL**.	**PCLK**	V_{SOC}
TDOSEL	Output	Selects between core and ETM **TDO**.	**PCLK**	V_{SOC}
THUMB2EN	Input	Indicates a Thumb-2 enabled core.	-	V_{CORE}
TMS	Input	Test mode select. Must be connected to the **TMS** input to the core.	See footnote [b]	V_{SOC}
TRACECLK	Output	Exported clock for **TRACEDATA[31:0]** and **TRACECTL**. Data is valid on both edges of this clock for maximum integrity.	-	V_{CORE}

Table A-1 ETM11CSSingle signals (continued)

Signal name	Input/ output	Description	Domains	
			Clock	Power
TRACECTL	Output	Used by trace capture devices. This signal is valid for the same time as **TRACEDATA**.Trigger = **TRACECTL** & !**TRACEDATA[0]** TraceDisabled = **TRACECTL** & **TRACEDATA[0]**.	**TRACECLK**	V_{SOC}
TRACEDATA[31:0]	Output	32-bit trace port. Only data on this bus must be captured.	**TRACECLK**	V_{SOC}
TRUSTZONEEN	Input	Indicates a TrustZone enabled core.	-	V_{CORE}

a. If **EXTSBYPASS** is 1, then **EXTIN[3:0]** must be generated on CLK and **EXTOUT[1:0]** must be sampled on **CLK**. If **EXTSBYPASS** is 0, then there is no such restriction. See *CoreSight ETM11 Integration Manual* for more details.

b. If **JTAGSBYPASS** is 1, then the JTAG interface inputs must be synchronous to PCLK. If **JTAGSBYPASS** is 0, then the JTAG inputs are synchronized internally.

A.2 ETM11CS Signals

Table A-2 lists the ETM11CS signals in alphabetical order. Clock domains, where specified, give the clock on which input signals must be generated and output signals sampled. See the *CoreSight ETM11 Integration Manual* for information about signals and connectivity.

Table A-2 ETM11CS signals

Signal	Input/ output	Description	Domains	
			Clock	Power
AFREADYM	Output	ATB interface FIFO flush finished.	**ATCLK**	V_{SOC}
AFVALIDM	Input	ATB interface FIFO flush request.	**ATCLK**	V_{SOC}
ASICCTL[7:0]	Output	Contents of the ASIC control register.	**CLK**	V_{CORE}
ATBYTESM[1:0]	Output	Size of **ATDATAM**.	**ATCLK**	V_{SOC}
ATCLK	Input	ATB interface clock.	-	V_{SOC}
ATCLKEN	Input	Enable signal for **ATCLK**.	**ATCLK**	V_{SOC}
ATDATAM[31:0]	Output	ATB interface data.	**ATCLK**	V_{SOC}
ATIDM[6:0]	Output	ATB interface trace source ID.	**ATCLK**	V_{SOC}
ATREADYM	Input	**ATDATA** can be accepted.	**ATCLK**	V_{SOC}
ATRESETn	Input	ATB interface reset.	Internally synchronized	V_{SOC}
ATVALIDM	Output	ATB interface data valid.	**ATCLK**	V_{SOC}
CFGBIGEND	Input	Output signal from ARM11 family cores. **CFGBIGEND** at logic 1 indicates that the core is operating in big-endian (BE-32) mode.	**CLK**	V_{SOC}
CLK	Input	This is the main clock for the ETM11CS.	-	V_{SOC}
CORESELECT[2:0]	Output	Where an ETM is shared between multiple cores, this signal specifies which core to trace.	-	V_{CORE}

Table A-2 ETM11CS signals (continued)

Signal	Input/ output	Description	Domains	
			Clock	Power
DBGACK	Input	Indicates that the core is in debug state. This signal is connected to the core general purpose **DBGACK** output, so that it can be used to determine when **ETMDBGRQ** can be deasserted. It is also used for other purposes in the ETM, and care must be taken to ensure the timing of this signal is appropriate because it does not come through the main core/ETM interface.	**CLK**	V_{SOC}
DBGEN	Input	Invasive debug enable. At logic 1, indicates that invasive debug is enabled.	-	V_{SOC}
ETMCPADDRESS[14:0]	Input	Coprocessor interface address. The relevant fields of the `MRC` or `MCR` instruction are encoded here as follows: Bits 14:12 <Opcode_1> Bits 11:8 <CRn> Bits 7:4 <CRm> Bit 3 = 0 cp14 Bit 3 = 1 cp15 Bits 2:0 <Opcode_2>.	**CLK**	V_{CORE}
ETMCPCOMMIT	Input	Coprocessor interface commit If this signal is LOW two cycles after **ETMCPENABLE** is asserted, the transfer is canceled and must not take any effect.	**CLK**	V_{CORE}
ETMCPENABLE	Input	Coprocessor interface enable **ETMCPWRITE** and **ETMCPADDRESS** are valid this cycle, and the remaining signals are valid two cycles later.	**CLK**	V_{CORE}
ETMCPSECCTL[1:0]	Input	Coprocessor interface security control	**CLK**	V_{CORE}
ETMCPWDATA[31:0]	Input	Coprocessor interface write value.	**CLK**	V_{CORE}
ETMCPWRITE	Input	Coprocessor interface read or write. This signal is asserted for writes.	**CLK**	V_{CORE}
ETMDA[31:3]	Input	Address for data transfer.	**CLK**	V_{CORE}

Table A-2 ETM11CS signals (continued)

Signal	Input/ output	Description	Domains	
			Clock	Power
ETMDACTL[17:0]	Input	Data address interface control signals.	**CLK**	V_{CORE}
ETMDBGRQ	Output	Request from the CoreSight ETM11 for the core to enter debug state. This must be ORed with any ASIC-level **DBGRQ** signals before being connected to the core **EDBGRQ** input.	**CLK**	V_{CORE}
ETMDD[63:0]	Input	Contains the data value for a Load, Store, `MRC`, or `MCR` instruction.	**CLK**	V_{CORE}
ETMDDCTL[3:0]	Input	Data interface control signals.	**CLK**	V_{CORE}
ETMDD2[63:0]	Input	Contains the data value for a Store or MCR instruction. Only for use with an MPCore processor.	**CLK**	V_{CORE}
ETMDDCTL2[1:0]	Input	Data interface control signals. Only for use with an ARM11 MPCore processor.	**CLK**	V_{CORE}
ETMEN	Output	When HIGH, the ATB interface is enabled. When LOW, logic related to the to the ATB interface can be clock-gated.	**CLK**	V_{CORE}
ETMIA[31:0]	Input	Address for executed instruction.	**CLK**	V_{CORE}
ETMIACTL[17:0]	Input	Instruction address interface control signals.	**CLK**	V_{CORE}
ETMIARET[31:0]	Input	Address to return to if branch is incorrectly predicted.	**CLK**	V_{CORE}
ETMIASECCTL[1:0]	Input	Instruction interface security control signals.	**CLK**	V_{CORE}
ETMPADV[2:0]	Input	Pipeline advance signals.	**CLK**	V_{CORE}

Table A-2 ETM11CS signals (continued)

Signal	Input/ output	Description	Domains	
			Clock	Power
ETMPWRUP	Output	When HIGH, indicates that CoreSight ETM11 is in use. When LOW: • external logic supporting CoreSight ETM11 can be clock-gated to conserve power • the ARM11 processor disables the CoreSight ETM11 interface • logic within CoreSight ETM11 is clock-gated to conserve power.	**CLK**	V_{CORE}
ETMWFIPENDING	Input	Indicates that the ARM11 processor is about to go into Standby mode, and that the ETM must drain its FIFO.	**CLK**	V_{CORE}
ETPSUP	Input	Indicates a trace port is connected.	-	V_{CORE}
EVNTBUS[19:0]	Input	Gives the status of the performance monitoring events. Used as extended external inputs. Not used on ARM11 MPCore processor.	**CLK**	V_{CORE}
EXTIN[3:0]	Input	External input resources.	See footnote [a]	V_{CORE}
EXTINACK[3:0]	Output	Acknowledge signal for the **EXTIN[3:0]** bus. When **EXTSBYPASS** is HIGH, **EXTINACK[3:0]** are not valid and can be ignored.	-	V_{CORE}
EXTOUT[1:0]	Output	External outputs.	See footnote [a]	V_{CORE}
EXTOUTACK[1:0]	Input	Acknowledge signals for the **EXTOUT[1:0]** bus. When **EXTSBYPASS** is HIGH, these signals must be tied LOW.	-	V_{CORE}

Table A-2 ETM11CS signals (continued)

Signal	Input/ output	Description	Domains	
			Clock	Power
EXTSBYPASS	Input	**EXTSBYPASS** is a single bit input to the ETM that is used to bypass the synchronization of the external inputs **EXTIN[3:0]** and external outputs **EXTOUT[1:0]**. When **EXTSBYPASS** is LOW: • a HIGH output on any bit of **EXTOUT[1:0]** is held until the corresponding bit of **EXTOUTACK[1:0]** is asserted HIGH • a HIGH on any bit of **EXTIN[3:0]** must remain held until the ETM asserts the corresponding bit of **EXTINACK[3:0]** HIGH. When **EXTSBYPASS** is HIGH, **EXTOUT[1:0]** outputs are valid on the rising edge of **CLK** and **EXTIN[3:0]** inputs must be valid on the rising edge of **CLK**.	-	V_{CORE}
FIFOPEEK[8:0]	Output	For validation purposes only. Indicates when various events occur before being written to the FIFO.	**CLK**	V_{CORE}
MAXCORES[2:0]	Input	Where an ETM is shared between multiple cores, this signal specifies the number of cores the ETM can trace. It must be tied to the number of cores sharing the ETM minus 1.	-	V_{CORE}
MAXEXTIN[2:0]	Input	External inputs supported by the ASIC (maximum 4). This appears in the configuration code register.	-	V_{CORE}
MAXEXTOUT[1:0]	Input	External outputs supported by the ASIC (maximum 2). This appears in the configuration code register.	-	V_{CORE}
MAXPORTSIZE[3:0]	Input	Maximum port size supported. See *MAXPORTSIZE in ETM11CS and ETM11CSSingle* on page 2-30.	-	V_{CORE}
nCORECLAMP	Input	Indicates power down of the Core domain. Active LOW.	-	V_{SOC}

Table A-2 ETM11CS signals (continued)

Signal	Input/ output	Description	Domains	
			Clock	Power
nETMWFIREADY	Output	Indicates that CoreSight ETM11 FIFO is empty and that the ARM11 processor can be put into Standby mode.	**CLK**	V_{CORE}
NIDEN	Input	Non-invasive debug enable.	-	V_{SOC}
nPORESET	Input	Main reset. Resets all registers in the CLK domain	Internally synchronized	V_{SOC}
nSOCCLAMP	Input	Indicates power down of the SoC domain. Active LOW.	-	V_{SOC}
PADDRDBG[11:2]	Input	Debug APB Address Bus.	**PCLKDBG**	V_{SOC}
PADDRDBG31	Input	Originates as an output signal from either the ETMJTAGPORT or *Debug Access Por*t (DAP): • **PADDRDBG31** at logic 1 indicates an access from hardware (JTAG) • **PADDRDBG31** at logic 0 indicates an access from software.	**PCLKDBG**	V_{SOC}
PCLKDBG	Input	Debug APB clock.	-	V_{SOC}
PCLKENDBG	Input	Debug APB clock enable.	**PCLKDBG**	V_{SOC}
PENABLEDBG	Input	The Debug APB interface is enabled for a transfer.	**PCLKDBG**	V_{SOC}
PORTMODE[2:0]	Output	Currently requested port mode. Not used in a CoreSight system.	**CLK**	V_{CORE}
PORTSIZE[3:0]	Output	Currently requested port size. Not used in a CoreSight system.	**CLK**	V_{CORE}
PRDATADBG[31:0]	Output	Debug APB read data.	**PCLKDBG**	V_{SOC}
PREADYDBG	Output	Used to extend Debug APB transfers.	**PCLKDBG**	V_{SOC}
PRESETDBGn	Input	Debug APB interface reset.	Internally synchronized	V_{SOC}
PSELDBG	Input	Debug APB Slave select signal.	**PCLKDBG**	V_{SOC}
PSLVERRDBG	Output	Debug APB error signal.	**PCLKDBG**	V_{SOC}

Table A-2 ETM11CS signals (continued)

Signal	Input/ output	Description	Domains	
			Clock	Power
PWDATADBG[31:0]	Input	Debug APB write data.	**PCLKDBG**	V_{SOC}
PWRITEDBG	Input	Debug APB transfer direction (!Read/Write).	**PCLKDBG**	V_{SOC}
RSTBYPASS	Input	Reset synchronization bypass.	-	V_{SOC}
SE	Input	Scan enable.	-	V_{SOC}
THUMB2EN	Input	Indicates a Thumb-2 enabled core.	**CLK**	V_{CORE}
TRIGOUT	Output	Trigger request status signal.	See footnote [b]	V_{SOC}
TRIGOUTACK	Input	ATB trigger acknowledge.	-	V_{SOC}
TRIGSBYPASS	Input	Trigger synchronization bypass.	-	V_{SOC}
TRUSTZONEEN	Input	Indicates a TrustZone enabled core.	**CLK**	V_{CORE}

a. If **EXTSBYPASS** is 1, then **EXTIN[3:0]** must be generated on CLK and **EXTOUT[1:0]** must be sampled on **CLK**. If **EXTSBYPASS** is 0, then there is no such restriction. See *CoreSight ETM11 Integration Manual* for further details.
b. If **TRIGSBYPASS** is 1, then **TRIGOUT** must be sampled on ATCLK. If **TRIGSBYPASS** is 0, then there is no such restriction.

Appendix B
I/O Signal Timings

This appendix lists and describes the CoreSight ETM11 I/O timing. It contains the following sections:

- *ETM11CSSingle I/O timing parameters* on page B-2
- *ETM11CS I/O timing parameters* on page B-6.

B.1 ETM11CSSingle I/O timing parameters

Signals are classified according to the percentage of the clock period taken up by internal logic.

- For inputs this is the delay between the input port and the first register.
- For outputs this is the delay between the last register and the output port.

The timing classifications used are:

Early Less than 20% of the period described above.

Middle Between 20% and 80% of the period described above.

Late Greater than 80% of the period described above.

Table B-1 describes the ETM11CSSingle signal timing parameters.

Table B-1 ETM11CSSingle signal timing parameters

Signal name	Timing classification	Input/ Output
ASICCTL[7:0]	Middle	Output
ATCLK	-	Input
ATCLKEN	Middle	Input
ATRESETn	Late	Input
CFGBIGEND	Middle	Input
CLK	-	Input
CORESELECT[2:0]	Middle	Output
DBGACK	Middle	Input
DBGEN	Middle	Input
ETBTRACEDATA[31:0]	Middle	Output
ETBTRACEVALID	Middle	Output
ETBTRIGGER	Middle	Output
ETMCPADDRESS[14:0]	Middle	Input
ETMCPCOMMIT	Middle	Input
ETMCPENABLE	Middle	Input
ETMCPSECCTL[1:0]	Middle	Input

Table B-1 ETM11CSSingle signal timing parameters (continued)

Signal name	Timing classification	Input/ Output
ETMCPWDATA[31:0]	Middle	Input
ETMCPWRITE	Middle	Input
ETMDA[31:3]	Middle	Input
ETMDACTL[17:0]	Middle	Input
ETMDBGRQ	Middle	Output
ETMDD[63:0]	Middle	Input
ETMDDCTL[3:0]	Middle	Input
ETMEN	Middle	Output
ETMIA[31:0]	Middle	Input
ETMIACTL[17:0]	Middle	Input
ETMDD2[63:0]	Middle	Input
ETMDDCTL2[1:0]	Middle	Input
ETMIARET[31:0]	Middle	Input
ETMIASECCTL[1:0]	Middle	Input
ETMPADV[2:0]	Middle	Input
ETMPWRUP	Middle	Output
ETMWFIPENDING	Middle	Input
EVNTBUS[19:0]	Middle	Input
EXTIN[3:0]	Middle	Input
EXTINACK[3:0]	Middle	Output
EXTOUT[1:0]	Middle	Output
EXTOUTACK[1:0]	Middle	Input
EXTSBYPASS	Middle	Input
FIFOPEEK[8:0]	Middle	Output

Table B-1 ETM11CSSingle signal timing parameters (continued)

Signal name	Timing classification	Input/ Output
JTAGSBYPASS	Late	Input
MAXCORES[2:0]	Middle	Input
MAXEXTIN[2:0]	Middle	Input
MAXEXTOUT[1:0]	Middle	Input
MAXPORTSIZE[3:0]	Middle	Input
nCORECLAMP	Late	Input
nETMWFIREADY	Middle	Output
NIDEN	Middle	Input
nPORESET	Late	Input
nSOCCLAMP	Late	Input
nTRST	Middle	Input
PADDR[11:2]	Late	Input
PCLK	-	Input
PCLKEN	Late	Input
PENABLE	Late	Input
PORTMODE[2:0]	Middle	Output
PORTSIZE[3:0]	Middle	Output
PRDATA[31:0]	Late	Output
PREADY	Late	Output
PRESETn	Late	Input
PSEL	Late	Input
PSLVERR	Late	Output
PWDATA[31:0]	Late	Input
PWRITE	Late	Input

Table B-1 ETM11CSSingle signal timing parameters (continued)

Signal name	Timing classification	Input/ Output
RSTBYPASS	Late	Input
RTCK	Middle	Output
SE	Late	Input
TCK	-	Input
TCKEN	Middle	Input
TDI	Late	Input
TDO	Middle	Output
TDOSEL	Middle	Output
THUMB2EN	Late	Input
TMS	Late	Input
TRACECLK	Middle	Output
TRACECTL	Middle	Output
TRACEDATA[31:0]	Middle	Output
TRUSTZONEEN	Late	Input

Note

Actual clock frequencies and input and output timing constraints vary according to application requirements and the silicon process technologies used. The maximum operating clock frequencies change according to the constraints and the process technology you use.

B.2 ETM11CS I/O timing parameters

Signals are classified according to the percentage of the clock period taken up by internal logic.

- For inputs this is the delay between the input port and the first register.
- For outputs this is the delay between the last register and the output port.

The timing classifications used are:

Early Less than 20% of the period described above.

Middle Between 20% and 80% of the period described above.

Late Greater than 80% of the period described above.

Table B-2 describes the ETM11CS signal timing parameters.

Table B-2 ETM11CS signal timing parameters

Signal name	Timing classification	Input/ Output
AFREADYM	Middle	Output
AFVALIDM	Middle	Input
ASICCTL[7:0]	Middle	Output
ATBYTESM[1:0]	Middle	Output
ATCLK	-	Input
ATCLKEN	Middle	Input
ATDATAM[31:0]	Middle	Output
ATIDM[6:0]	Middle	Input
ATREADYM	Middle	Input
ATRESETn	Late	Input
ATVALIDM	Middle	Output
CFGBIGEND	Middle	Input
CLK	-	Input
CORESELECT[2:0]	Middle	Output
DBGACK	Middle	Input
DBGEN	Middle	Input

Table B-2 ETM11CS signal timing parameters (continued)

Signal name	Timing classification	Input/ Output
ETMCPADDRESS[14:0]	Middle	Input
ETMCPCOMMIT	Middle	Input
ETMCPENABLE	Middle	Input
ETMCPSECCTL[1:0]	Middle	Input
ETMCPWDATA[31:0]	Middle	Input
ETMCPWRITE	Middle	Input
ETMDA[31:3]	Middle	Input
ETMDACTL[17:0]	Middle	Input
ETMDD2[63:0]	Middle	Input
ETMDDCTL2[1:0]	Middle	Input
ETMDBGRQ	Middle	Output
ETMDD[63:0]	Middle	Input
ETMDDCTL[3:0]	Middle	Input
ETMEN	Middle	Output
ETMIA[31:0]	Middle	Input
ETMIACTL[17:0]	Middle	Input
ETMIARET[31:0]	Middle	Input
ETMIASECCTL[1:0]	Middle	Input
ETMPADV[2:0]	Middle	Input
ETMPWRUP	Middle	Output
ETMWFIPENDING	Middle	Input
ETPSUP	Late	Input
EVNTBUS[19:0]	Middle	Input
EXTIN[3:0]	Middle	Input

Table B-2 ETM11CS signal timing parameters (continued)

Signal name	Timing classification	Input/ Output
EXTINACK[3:0]	Middle	Output
EXTOUT[1:0]	Middle	Output
EXTOUTACK[1:0]	Middle	Input
EXTSBYPASS	Middle	Input
FIFOPEEK[8:0]	Middle	Output
MAXCORES[2:0]	Middle	Input
MAXEXTIN[2:0]	Middle	Input
MAXEXTOUT[1:0]	Middle	Input
MAXPORTSIZE[3:0]	Middle	Input
nCORECLAMP	Late	Input
nETMWFIREADY	Middle	Output
NIDEN	Middle	Input
nPORESET	Late	Input
nSOCCLAMP	Late	Input
PADDRDBG[11:2]	Middle	Input
PADDRDBG31	Middle	Input
PCLKDBG	Middle	Input
PCLKENDBG	Middle	Input
PENABLEDBG	Middle	Input
PORTMODE[2:0]	Middle	Output
PORTSIZE[3:0]	Middle	Output
PRDATADBG[31:0]	Middle	Output
PREADYDBG	Middle	Output
PRESETDBGn	Late	Input

Table B-2 ETM11CS signal timing parameters (continued)

Signal name	Timing classification	Input/ Output
PSELDBG	Middle	Input
PSLVERRDBG	Middle	Output
PWDATADBG[31:0]	Middle	Input
PWRITEDBG	Middle	Input
RSTBYPASS	Late	Input
SE	Late	Input
THUMB2EN	Late	Input
TRIGOUT	Middle	Output
TRIGOUTACK	Middle	Input
TRIGSBYPASS	Middle	Input
TRUSTZONEEN	Late	Input

Note

Actual clock frequencies and input and output timing constraints vary according to application requirements and the silicon process technologies used. The maximum operating clock frequencies change according to the constraints and the process technology you use.

Glossary

This glossary describes some of the terms used in technical documents from ARM Limited.

Advanced High-performance Bus (AHB)
A bus protocol with a fixed pipeline between address/control and data phases. It only supports a subset of the functionality provided by the AMBA AXI protocol. The full AMBA AHB protocol specification includes a number of features that are not commonly required for master and slave IP developments and ARM Limited recommends only a subset of the protocol is usually used. This subset is defined as the AMBA AHB-Lite protocol.

See also Advanced Microcontroller Bus Architecture and AHB-Lite.

Advanced Microcontroller Bus Architecture (AMBA)
A family of protocol specifications that describe a strategy for the interconnect. AMBA is the ARM open standard for on-chip buses. It is an on-chip bus specification that details a strategy for the interconnection and management of functional blocks that make up a *System-on-Chip* (SoC). It aids in the development of embedded processors with one or more CPUs or signal processors and multiple peripherals. AMBA complements a reusable design methodology by defining a common backbone for SoC modules.

See also Advanced High-performance Bus and AHB-Lite.

Advanced Peripheral Bus (APB)
: A simpler bus protocol than AXI and AHB. It is designed for use with ancillary or general-purpose peripherals such as timers, interrupt controllers, UARTs, and I/O ports. Connection to the main system bus is through a system-to-peripheral bus bridge that helps to reduce system power consumption.

  See also Advanced High-performance Bus.

AHB
: *See* Advanced High-performance Bus.

AHB-Lite
: A subset of the full AMBA AHB protocol specification. It provides all of the basic functions required by the majority of AMBA AHB slave and master designs, particularly when used with a multi-layer AMBA interconnect. In most cases, the extra facilities provided by a full AMBA AHB interface are implemented more efficiently by using an AMBA AXI protocol interface.

Aligned
: A data item stored at an address that is divisible by the number of bytes that defines the data size is said to be aligned. Aligned words and halfwords have addresses that are divisible by four and two respectively. The terms word-aligned and halfword-aligned therefore stipulate addresses that are divisible by four and two respectively.

AMBA
: *See* Advanced Microcontroller Bus Architecture.

APB
: *See* Advanced Peripheral Bus.

Architecture
: The organization of hardware and/or software that characterizes a processor and its attached components, and enables devices with similar characteristics to be grouped together when describing their behavior, for example, Harvard architecture, instruction set architecture, ARMv6 architecture.

ARM state
: A processor that is executing ARM (32-bit) word-aligned instructions is operating in ARM state.

  See also Thumb state.

Big-endian
: Byte ordering scheme in which bytes of decreasing significance in a data word are stored at increasing addresses in memory.

  See also Little-endian and Endianness.

Branch folding
: Branch folding is a technique where, on the prediction of most branches, the branch instruction is completely removed from the instruction stream presented to the execution pipeline. Branch folding can significantly improve the performance of branches, taking the CPI for branches below one.

Branch prediction
: The process of predicting if conditional branches are to be taken or not in pipelined processors. Successfully predicting if branches are to be taken enables the processor to prefetch the instructions following a branch before the condition is fully resolved.

Branch prediction can be done in software or by using custom hardware. Branch prediction techniques are categorized as static, in which the prediction decision is decided before run time, and dynamic, in which the prediction decision can change during program execution.

Breakpoint A breakpoint is a mechanism provided by debuggers to identify an instruction at which program execution is to be halted. Breakpoints are inserted by the programmer to enable inspection of register contents, memory locations, variable values at fixed points in the program execution to test that the program is operating correctly. Breakpoints are removed after the program is successfully tested.

See also Watchpoint.

Burst A group of transfers to consecutive addresses. Because the addresses are consecutive, there is no requirement to supply an address for any of the transfers after the first one. This increases the speed at which the group of transfers can occur. Bursts over AMBA are controlled using signals to indicate the length of the burst and how the addresses are incremented.

Byte An 8-bit data item.

Byte invariant In a byte-invariant system, the address of each byte of memory remains unchanged when switching between little-endian and big-endian operation. When a data item larger than a byte is loaded from or stored to memory, the bytes making up that data item are arranged into the correct order depending on the endianness of the memory access. The ARM architecture supports byte-invariant systems in ARMv6 and later versions. When byte-invariant support is selected, unaligned halfword and word memory accesses are also supported. Multi-word accesses are expected to be word-aligned.

See also Word-invariant.

Byte lane strobe A signal that is used for unaligned or mixed-endian data accesses to determine which byte lanes are active in a transfer. One bit of this signal corresponds to eight bits of the data bus.

Clock gating Gating a clock signal for a macrocell with a control signal, and using the modified clock that results to control the operating state of the macrocell.

Cold reset Also known as power-on reset. Starting the processor by turning power on. Turning power off and then back on again clears main memory and many internal settings. Some program failures can lock up the processor and require a cold reset to enable the system to be used again. In other cases, only a warm reset is required.

See also Warm reset.

Coprocessor A processor that supplements the main processor. It carries out additional functions that the main processor cannot perform. Usually used for floating-point math calculations, signal processing, or memory management.

Core A core is that part of a processor that contains the ALU, the datapath, the general-purpose registers, the Program Counter, and the instruction decode and control circuitry.

Core reset *See* Warm reset.

CoreSight The infrastructure for monitoring, tracing, and debugging a complete system on chip.

CPI *See* Cycles per instruction.

Cycles Per instruction (CPI)
Cycles per instruction (or clocks per instruction) is a measure of the number of computer instructions that can be performed in one clock cycle. This figure of merit can be used to compare the performance of different CPUs that implement the same instruction set against each other. The lower the value, the better the performance.

DAP *See* Debug Access Port.

Debug Access Port (DAP)
A TAP block that acts as an AMBA, AHB or AHB-Lite, master for access to a system bus. The DAP is the term used to encompass a set of modular blocks that support system wide debug. The DAP is a modular component, intended to be extendable to support optional access to multiple systems such as memory mapped AHB and CoreSight APB through a single debug interface.

Doubleword A 64-bit data item. The contents are taken as being an unsigned integer unless otherwise stated.

EmbeddedICE logic An on-chip logic block that provides TAP-based debug support for ARM processor cores. It is accessed through the TAP controller on the ARM core using the JTAG interface.

EmbeddedICE-RT The JTAG-based hardware provided by debuggable ARM processors to aid debugging in real-time.

Embedded Trace Buffer (ETB)
The ETB provides on-chip storage of trace data using a configurable sized RAM.

Embedded Trace Macrocell (ETM)
A hardware macrocell that outputs instruction and data trace information on a trace port.

Endianness Byte ordering. The scheme that determines the order in which successive bytes of a data word are stored in memory.

See also Little-endian and Big-endian.

ETB *See* Embedded Trace Buffer.

ETM *See* Embedded Trace Macrocell.

Exception A fault or error event that is considered serious enough to require that program execution is interrupted. Examples include attempting to perform an invalid memory access, external interrupts, and undefined instructions. When an exception occurs, normal program flow is interrupted and execution is resumed at the corresponding exception vector. This contains the first instruction of the interrupt handler to deal with the exception.

Exception vector *See* Interrupt vector.

Fast context switch

In a multitasking system, the point at which the time-slice allocated to one process stops and the one for the next process starts. If processes are switched often enough, they can appear to a user to be running in parallel, in addition to being able to respond quicker to external events that might affect them.

In ARM processors, a fast context switch is caused by the selection of a non-zero PID value to switch the context to that of the next process. A fast context switch causes each Virtual Address for a memory access, generated by the ARM processor, to produce a Modified Virtual Address that is sent to the rest of the memory system to be used in place of a normal Virtual Address. For some cache control operations Virtual Addresses are passed to the memory system as data. In these cases no address modification takes place.

See also Fast Context Switch Extension.

Fast Context Switch Extension (FCSE)

An extension to the ARM architecture that enables cached processors with an MMU to present different addresses to the rest of the memory system for different software processes, even when those processes are using identical addresses.

See also Fast context switch.

FCSE *See* Fast Context Switch Extension.

Flat address mapping

A system of organizing memory in which each physical address contained within the memory space is the same as its corresponding virtual address.

Half-rate clocking Half-rate clocking is a feature of the ETM. It means dividing the trace clock by two so that the TPA can sample trace data signals on both the rising and falling edges of the trace clock. The primary purpose of half-rate clocking is to reduce the signal transition rate on the trace clock of an ASIC for very high-speed systems.

High vectors Alternative locations for exception vectors. The high vector address range is near the top of the address space, rather than at the bottom.

IEM *See* Intelligent Energy Management.

Implementation-defined
Behavior that is not architecturally defined, but is defined and documented by individual implementations.

Instruction cycle count
The number of cycles for which an instruction occupies the Execute stage of the pipeline.

Intelligent Energy Management (IEM)
A technology that enables dynamic voltage scaling and clock frequency variation to be used to reduce power consumption in a device.

Interrupt vector One of a number of fixed addresses in low memory, or in high memory if high vectors are configured, that contains the first instruction of the corresponding interrupt handler.

See also High vectors.

Little-endian Byte ordering scheme in which bytes of increasing significance in a data word are stored at increasing addresses in memory.

See also Big-endian and Endianness.

Monitor debug-mode
One of two mutually exclusive debug modes. In Monitor debug-mode the processor enables a software abort handler provided by the debug monitor or operating system debug task. When a breakpoint or watchpoint is encountered, this enables vital system interrupts to continue to be serviced while normal program execution is suspended.

See also Halt mode.

Power-on reset *See* Cold reset.

Prefetching In pipelined processors, the process of fetching instructions from memory to fill up the pipeline before the preceding instructions have finished executing. Prefetching an instruction does not mean that the instruction must be executed.

Processor A processor is the circuitry in a computer system required to process data using the computer instructions. It is an abbreviation of microprocessor. A clock source, power supplies, and main memory are also required to create a minimum complete working computer system.

Read Reads are defined as memory operations that have the semantics of a load. That is, the ARM instructions LDM, LDRD, LDC, LDR, LDRT, LDRSH, LDRH, LDRSB, LDRB, LDRBT, LDREX, RFE, STREX, SWP, and SWPB, and the Thumb instructions LDM, LDR, LDRSH, LDRH, LDRSB, LDRB, and POP.

Java instructions that are accelerated by hardware can cause a number of reads to occur, according to the state of the Java stack and the implementation of the Java hardware acceleration.

Reserved A field in a control register or instruction format is reserved if the field is to be defined by the implementation, or produces Unpredictable results if the contents of the field are not zero. These fields are reserved for use in future extensions of the architecture or are implementation-specific. All reserved bits not used by the implementation must be written as 0 and read as 0.

SBO *See* Should Be One.

SBZ *See* Should Be Zero.

SBZP *See* Should Be Zero or Preserved.

Scan chain A scan chain is made up of serially-connected devices that implement boundary scan technology using a standard JTAG TAP interface. Each device contains at least one TAP controller containing shift registers that form the chain connected between **TDI** and **TDO**, through which test data is shifted. Processors can contain several shift registers to enable you to access selected parts of the device.

Should Be One (SBO)
Should be written as 1 (or all 1s for bit fields) by software. Writing a 0 produces Unpredictable results.

Should Be Zero (SBZ)
Should be written as 0 (or all 0s for bit fields) by software. Writing a 1 produces Unpredictable results.

Should Be Zero or Preserved (SBZP)
Should be written as 0 (or all 0s for bit fields) by software, or preserved by writing back the same value that has been previously read from the same field on the same processor.

Synchronization primitive
The memory synchronization primitive instructions are instructions that are used to ensure memory synchronization, that is, the LDREX, STREX, SWP, and SWPB instructions.

TAP *See* Test Access Port.

Test Access Port (TAP)
The collection of four mandatory and one optional terminals that form the input/output and control interface to a JTAG boundary-scan architecture. The mandatory terminals are **TDI**, **TDO**, **TMS**, and **TCK**. The optional terminal is **TRST**. This signal is mandatory in ARM cores because it is used to reset the debug logic.

Thumb instruction
A halfword that specifies an operation for an ARM processor in Thumb state to perform. Thumb instructions must be halfword-aligned.

Thumb state
A processor that is executing Thumb (16-bit) halfword aligned instructions is operating in Thumb state.

TPA
See *Trace Port Analyzer*.

Trace Port Analyzer (TPA)
A hardware device that captures trace information output on a trace port. This can be a low-cost product designed specifically for trace acquisition, or a logic analyzer.

Undefined
Indicates an instruction that generates an Undefined instruction trap. See the *ARM Architectural Reference Manual* for more information on ARM exceptions.

UNP
See Unpredictable.

Unpredictable (UNP)
For reads, the data returned when reading from this location is unpredictable. It can have any value. For writes, writing to this location causes unpredictable behavior, or an unpredictable change in device configuration. Unpredictable instructions must not halt or hang the processor, or any part of the system.

In an ETM context, means that the behavior of the ETM cannot be relied on. Such conditions have not been validated. When applied to the programming of an event resource, only the output of that event resource is Unpredictable.

Unpredictable ETM behavior can affect the behavior of the entire system, because the ETM is capable of causing the core to enter debug state, and external outputs can be used for other purposes.

Warm reset
Also known as a core reset. Initializes the majority of the processor excluding the debug controller and debug logic. This type of reset is useful if you are using the debugging features of a processor.

Watchpoint
A watchpoint is a mechanism provided by debuggers to halt program execution when the data contained by a particular memory address is changed. Watchpoints are inserted by the programmer to enable inspection of register contents, memory locations, and variable values when memory is written to test that the program is operating correctly. Watchpoints are removed after the program is successfully tested.

See also Breakpoint.

Word-invariant

In a word-invariant system, the address of each byte of memory changes when switching between little-endian and big-endian operation, in such a way that the byte with address A in one endianness has address A EOR 3 in the other endianness. As a result, each aligned word of memory always consists of the same four bytes of memory in the same order, regardless of endianness. The change of endianness occurs because of the change to the byte addresses, not because the bytes are rearranged.

The ARM architecture supports word-invariant systems in ARMv3 and later versions. When word-invariant support is selected, the behavior of load or store instructions that are given unaligned addresses is instruction-specific, and is in general not the expected behavior for an unaligned access. It is recommended that word-invariant systems use the endianness that produces the desired byte addresses at all times, apart possibly from very early in their reset handlers before they have set up the endianness, and that this early part of the reset handler must use only aligned word memory accesses.

See also Byte-invariant.

Write

Writes are defined as operations that have the semantics of a store. That is, the ARM instructions SRS, STM, STRD, STC, STRT, STRH, STRB, STRBT, STREX, SWP, and SWPB, and the Thumb instructions STM, STR, STRH, STRB, and PUSH.

Java instructions that are accelerated by hardware can cause a number of writes to occur, according to the state of the Java stack and the implementation of the Java hardware acceleration.