CoreSight™ PTM-A9

Revision: r1p0

Technical Reference Manual

ARM DDI 0401C (ID073011)

CoreSight PTM-A9
Technical Reference Manual

Release Information

The following changes have been made to this book.

Change history

Date	Issue	Confidentiality	Change
11 April 2008	A	Non-Confidential	First release for r0p0
31 December 2008	B	Non-Confidential	First release for r1p0
08 July 2011	C	Non-Confidential	Update for r1p0

Proprietary Notice

Confidentiality Status

Product Status

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

Web Address

`http://www.arm.com`

Contents
CoreSight PTM-A9 Technical Reference Manual

Preface

This preface introduces the *CoreSight PTM-A9 Technical Reference Manual*. It contains the following sections:

- *About this book* on page v
- *Feedback* on page viii.

About this book

This is the *Technical Reference Manual* (TRM) for the *CoreSight PTM-A9 Program Trace Macrocell* (PTM). This manual describes the external functionality of the PTM.

Product revision status

The r*n*p*n* identifier indicates the revision status of the product described in this book, 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 book is written for system designers, system integrators, and verification engineers who are designing a *System-on-Chip* (SoC) device that uses the PTM.

Using this book

This book is organized into the following chapters:

Chapter 1 *Introduction*

Read this chapter for a general description of the PTM and its features.

Chapter 2 *Programmers Model*

Read this chapter for a description of the processor registers and programming details implemented in the PTM.

Appendix A *Signal Descriptions*

Read this appendix for a description of the PTM input and output signals.

Appendix B *Revisions*

Read this for a description of the technical changes between released issues of this book.

Glossary

The *ARM Glossary* is a list of terms used in ARM documentation, together with definitions for those terms. The *ARM Glossary* does not contain terms that are industry standard unless the ARM meaning differs from the generally accepted meaning.

The *ARM Glossary* is available on the ARM Infocenter at, `http://infocenter.arm.com/help/topic/com.arm.doc.aeg0014-/index.html`.

Conventions

Conventions that this book can use are described in:

- *Typographical*
- *Signals* on page vi.

Typographical

The typographical conventions are:

italic	Introduces special terminology, denotes 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.
`monospace`	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 >	Enclose replaceable terms for assembler syntax where they appear in code or code fragments. For example: `MRC p15, 0 <Rd>, <CRn>, <CRm>, <Opcode_2>`

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
- LOW for active-LOW signals.

Lower-case n At the start or end of a signal name denotes an active-LOW signal.

Additional reading

This section lists publications by ARM and by third parties.

See Infocenter, `http://infocenter.arm.com`, for access to ARM documentation.

ARM publications

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

- *CoreSight Program Flow Trace Architecture Specification* (ARM IHI 0035)
- *Embedded Trace Macrocell™ Architecture Specification* (ARM IHI 0114)
- *Cortex®-A9 Technical Reference Manual* (ARM DDI 0388)
- *Cortex-A9 Configuration and Sign-Off Guide* (ARM DII 00146)
- *Cortex-A9 MBIST TRM* (ARM DDI 0414)
- *CoreSight PTM-A9 Configuration and Sign-off Guide* (ARM DII 0161)
- *CoreSight PTM-A9 Integration Manual* (ARM DII 0162)
- *CoreSight Technology System Design Guide* (ARM DGI 0012)
- *CoreSight Components Technical Reference Manual* (ARM DDI 0314)
- *Cortex-A9 MP r0p0 Technical Reference Manual* (ARM DDI 0407)
- *Cortex-A9 Floating-Point Unit (FPU) Technical Reference Manual* (ARM DDI 0408)

- *Cortex-A9 NEON™ Media Processing Engine Technical Reference Manual* (ARM DDI 0409)
- *AMBA® 3 APB Protocol Specification* (ARM IHI 0024)
- *RealView® Compilation Tools Developer Guide* (ARM DUI 0203)
- *RealView ICE and RealView Trace User Guide* (ARM DUI 0155)
- *Intelligent Energy Controller Technical Overview* (ARM DTO 0005)
- *ARM Architecture Reference Manual, ARMv7-A and ARMv7-R edition* (ARM DDI 0406).

Feedback

ARM welcomes feedback on this product and its documentation.

Feedback on this product

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

- The product name.
- The product revision or version.
- An explanation with as much information as you can provide. Include symptoms and diagnostic procedures if appropriate.

Feedback on content

If you have comments on content then send an e-mail to `errata@arm.com`. Give:

- the title
- the number, ARM DDI 0401C
- the page numbers to which your comments apply
- a concise explanation of your comments.

ARM also welcomes general suggestions for additions and improvements.

Note

ARM tests the PDF only in Adobe Acrobat and Acrobat Reader, and cannot guarantee the quality of the represented document when used with any other PDF reader.

Chapter 1
Introduction

This chapter introduces the *Program Flow Trace Macrocell* (PTM) for the Cortex-A9 processor. It contains the following sections:

1.1 About the PTM

The PTM is a module that performs real-time instruction flow tracing based on the *Program Flow Trace* (PFT) architecture. The PTM generates information that trace tools use to reconstruct the execution of all or part of a program.

The PFT architecture assumes that the trace tools can access a copy of the code being traced. For this reason, the PTM generates trace only at certain points in program execution, called *waypoints*. This reduces the amount of trace data generated by the PTM compared to the ETM protocol. Waypoints are changes in the program flow or events, such as an exception. The trace tools use waypoints to follow the flow of program execution.

For full reconstruction of the program flow, the PTM traces:

- indirect branches, with target address and condition code
- direct branches with only the condition code
- instruction barrier instructions
- exceptions, with an indication of where the exception occurred
- changes in processor instruction set state
- changes in processor security state
- context-ID changes
- entry to and return from Debug state when Halting Debug-mode is enabled.

You can also configure the PTM to trace:

- cycle count between traced waypoints
- global system timestamps
- target addresses for taken direct branches.

The PTM is a CoreSight™ component, and is an integral part of the ARM Real-time Debug solution, RealView®. For more information about CoreSight, see the CoreSight documentation listed in *Additional reading* on page vi. For full details of the PFT architecture, see the *Program Flow Trace Architecture Specification*.

1.1.1 PTM components

The PTM contains the following main components:

Processor interface

This interface monitors the behavior of the processor. The trace interface on the processor is described in the *Cortex-A9 Technical Reference Manual*.

Trace generation

This component generates a real-time trace stream.

Filtering and triggering resources

This component consists of various resources that can be used to affect when trace is generated, and to control the capturing of trace by the trace tools. It can use resources together with the cross-trigger matrix to link system debug and trace components together.

Main FIFO This component flattens out any bursts in the trace stream. If the FIFO becomes full, it signals an overflow in the trace. The trace generation logic does not generate any new trace until the FIFO empties.

AMBA 3 ATB interface

The PTM outputs trace using the Advanced Microcontroller Bus Architecture (AMBA) 3 *Advanced Trace Bus* (ATB) interface. See the *CoreSight Architecture Specification* for more information on AMBA 3 ATB.

The PTM can output trace asynchronously to the processor clock.

AMBA 3 APB interface

The AMBA 3 *Advanced Peripheral Bus* (APB) interface controls access to the PTM registers. See the *AMBA 3 APB Protocol Specification*.

1.1.2 The CoreSight debug environment

The PTM is designed for use with CoreSight, an extensible, system-wide debug and trace architecture from ARM.

A software debugger provides the user interface to the PTM. You can use this interface to configure the PTM facilities such as filtering and optional parts of the trace, such as timestamping. The same user interface is used to configure the other CoreSight components such as the *Trace Port Interface Unit* (TPIU), and to access the processor debug and performance monitor units. The debugger also interprets the trace information that has been output.

In addition, a CoreSight system can provide memory mapped access from the processor to its own debug and trace components.

The PTM outputs its trace stream to the AMBA 3 ATB interface. The CoreSight infrastructure provides the following options:

- Export the trace information through a trace port. An external *Trace Port Analyzer* (TPA) captures the trace information as Figure 1-1 on page 1-4 shows.
- Write the trace information directly to an on-chip *Embedded Trace Buffer* (ETB). You can read out the trace at low speed using a JTAG or Serial Wire interface when the trace capture is complete as Figure 1-1 on page 1-4 shows.

The debugger extracts the captured trace data from the TPA or ETB and decompresses it to provide full disassembly, with symbols, of the code that was executed. The PTM provides to the debugger the capability to link this data back to the original high-level source code, to provide a visualization of how the code was executed on the target system.

The PTM provides a mechanism for inserting a global timestamp in the trace stream that it generates. You can use this as an alternative or in addition to the cycle counting function. Trace tools can use timestamps to correlate trace from several trace sources. To use the timestamp function, the system integrator must provide a timestamp count to all trace sources that generate timestamps.

Figure 1-1 on page 1-4 shows how the PTM fits into the CoreSight debug environment. See the *CoreSight Architecture Specification* for more information.

In Figure 1-1 on page 1-4, the PTM and the *Cross Trigger Interface* (CTI), are part of a CoreSight system consisting of other processors with PTMs, and various other trace sources. You can program the CoreSight components using the *Debug Access Port* (DAP) through the CoreSight Debug APB bus. The system outputs trace over the ATB trace bus, and either exports it through the *Trace Port Interface Unit* (TPIU), or stores it in the *Embedded Trace Buffer* (ETB). Figure 1-1 on page 1-4 shows the trace sources connecting to both a TPIU and the ETB.

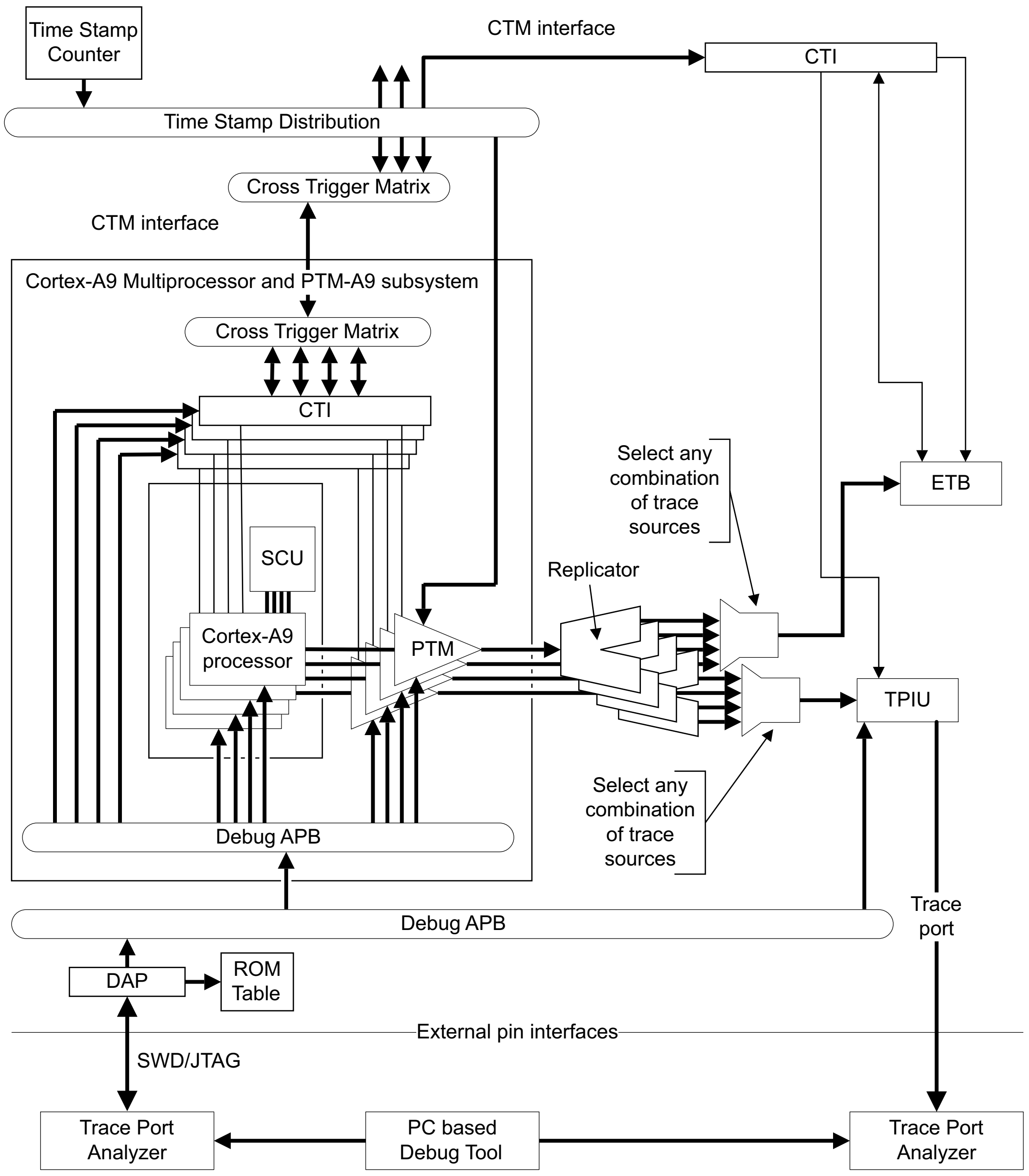

Figure 1-1 Example CoreSight debug environment

1.2 PTM configuration

Table 1-1 shows the options implemented in the PTM.

Table 1-1 PTM implementation options

Resource	Implemented, or number of instances
Number of address comparator pairs	4
Context ID comparators	1
Embedded ICE watchpoint inputs	0
Counters	2
Sequencers	1
External inputs	4
External outputs	2
Extended external inputs, PMUEVENT	52
Extended external input selectors	2
Instrumentation resources	0
FIFOFULL supported	No
Software access to registers?	Yes
FIFO depth	72 bytes
Trace output	ATB bus

1.3 PTM components

Figure 1-2 shows the main functional blocks of the PTM.

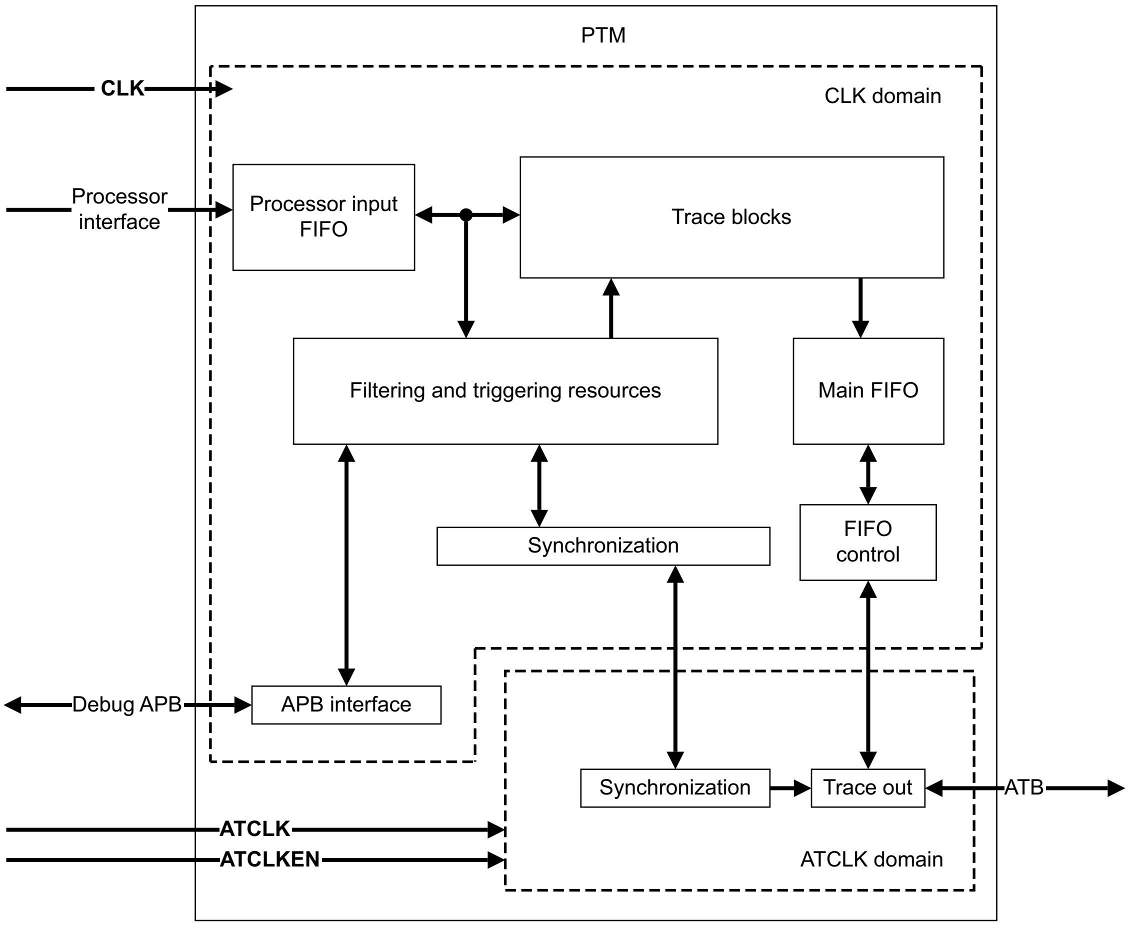

Figure 1-2 PTM functional blocks

1.3.1 Processor input FIFO

This block buffers the output from the processor trace interface until the execution of each waypoint is confirmed by the processor.

1.3.2 Trace blocks

These blocks generate the compressed trace based on the waypoint signals provided by the Cortex-A9 processor and the trace control signals from the resource blocks. This trace is passed to the FIFO.

1.3.3 Main FIFO

These blocks provide a buffer for bursts of trace data and manage the transfer of trace data into the **ATCLK** domain.

1.3.4 Clock domains

The PTM has its own clock domain, **CLK**, that must be synchronous with the processor clock. This clock domain is asynchronous to the **ATCLK** domain.

1.3.5 Level shifters

If you use *Intelligent Energy Management* (IEM) with the PTM, you can add level shifters during implementation. Level shifters shift the voltage level for the synchronization logic crossing between voltage domains.

1.3.6 Cross-Trigger Interface

The PTM enables cross-triggering using CTIs connected together through a *Cross Trigger Matrix* (CTM). Figure 1-3 and Figure 1-4 on page 1-8 show how each CTI is connected to its corresponding processor and PTM. This connection includes the ability for a debugger to synchronize debug entry and exit between multiple processors.

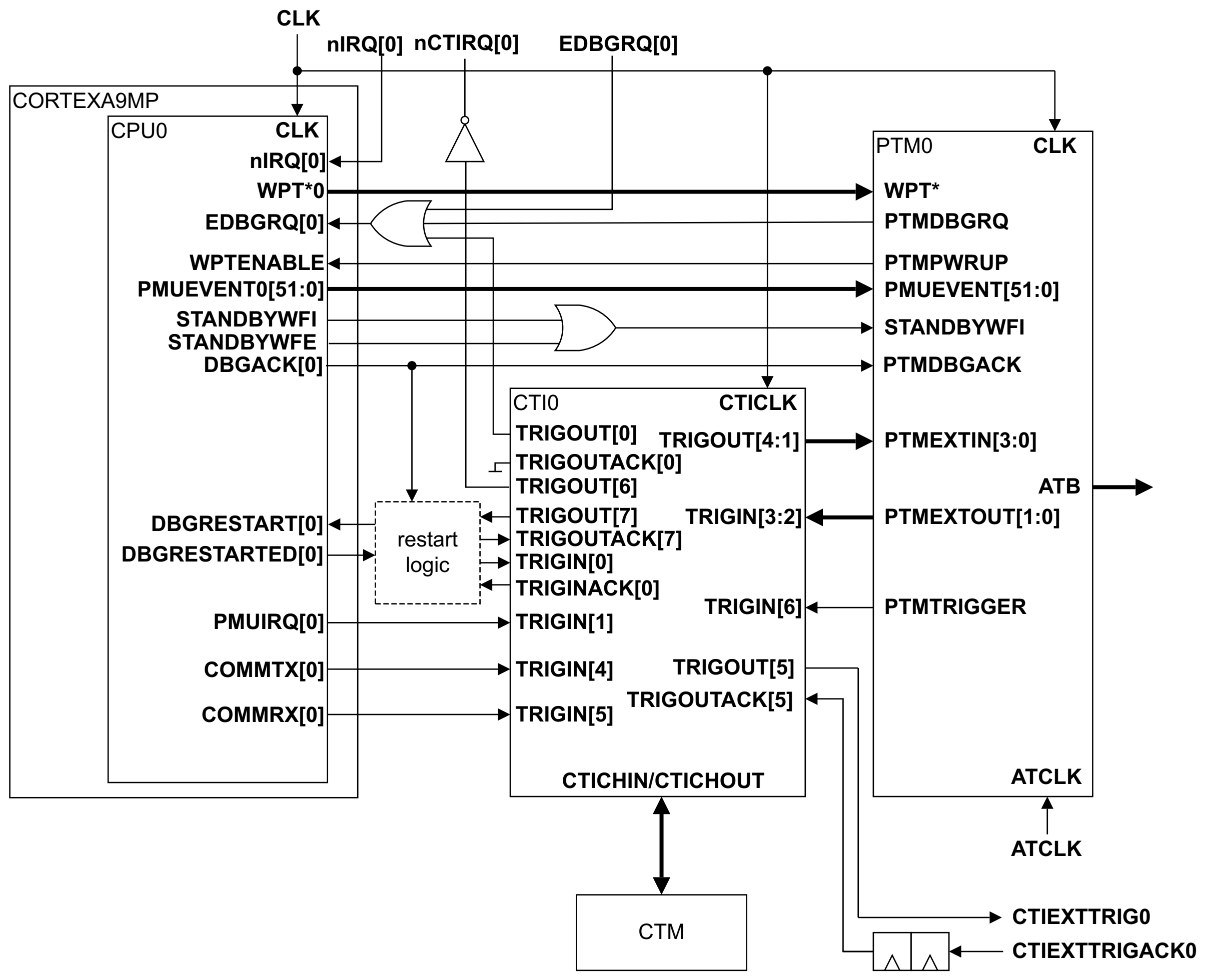

Figure 1-3 Cortex-A9 CTI connections

Figure 1-4 on page 1-8 shows the Cortex-A9 connections specific to debug request and restart.

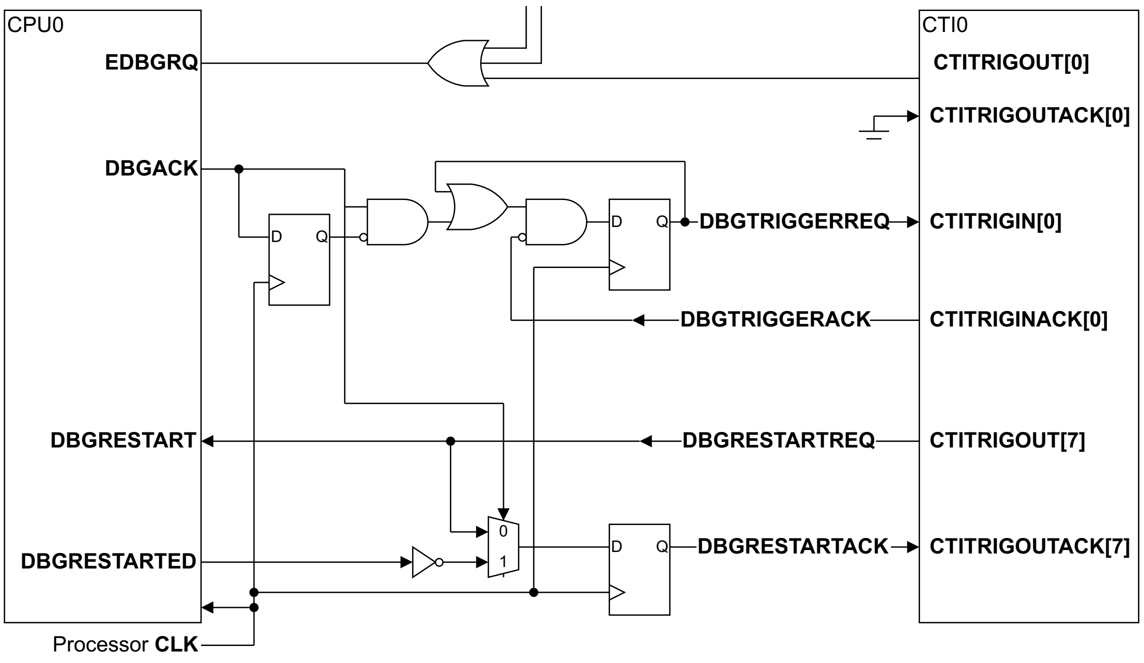

Figure 1-4 Cortex-A9 debug request and restart-specific connections

1.4 Prohibited regions for trace

The Cortex-A9 processor might prohibit tracing for some regions of code. When the processor is executing code from a prohibited region, the PTM does not output any information about program execution. The PTM resumes tracing when the processor leaves the prohibited region, see the *Program Flow Trace Architecture Specification*.

To decide whether an instruction can be traced, the processor examines:

- the current security level
- the current privilege level
- the state of the **SPNIDEN** and **SPIDEN** input pins
- the state of the **SUNIDEN** control bit set using `MCR p15, 0, Rd, c1, c0, 0`.

Table 1-2 shows the rules that determine whether trace is prohibited.

Table 1-2 Prohibited region decision table

Security level	Privilege level	NIDEN	DBGEN	SPNIDEN	SPIDEN	SUNIDEN	Prohibited
x	x	LOW	LOW	x	x	x	Yes
Non-secure	x	HIGH	x	x	x	x	No
Non-secure	x	x	HIGH	x	x	x	No
Secure	User	HIGH	x	x	x	Set	No
Secure	User	x	HIGH	x	x	Set	No
Secure	User	HIGH	x	LOW	LOW	Clear	Yes
Secure	User	x	HIGH	LOW	LOW	Clear	Yes
Secure	User	HIGH	x	HIGH	x	Clear	No
Secure	User	x	HIGH	HIGH	x	Clear	No
Secure	User	HIGH	x	x	HIGH	Clear	No
Secure	User	x	HIGH	x	HIGH	Clear	No
Secure	Privileged	HIGH	x	LOW	LOW	x	Yes
Secure	Privileged	x	HIGH	LOW	LOW	x	Yes
Secure	Privileged	HIGH	x	HIGH	x	x	No
Secure	Privileged	x	HIGH	HIGH	x	x	No
Secure	Privileged	HIGH	x	x	HIGH	x	No
Secure	Privileged	x	HIGH	x	HIGH	x	No

This elseif statement combines the decision table elements:

```
If (~NIDEN and ~DBGEN)
     Prohibited = 1
else if (Security level == Non-secure)
     Prohibited = 0
else if ((Privilege level == user) and (SUNIDEN))
     Prohibited = 0
else if (SPIDEN or SPNIDEN)
```

```
     Prohibited = 0
else
     Prohibited = 1
```

1.5 Reset behavior

The Main Control Register, `0x00`, is reset only on a power-on reset, **nPTMRESET** LOW. See Chapter 2 *Programmers Model* for a description of the reset values for specific registers.

The resets for the processor and the PTM are usually separate to permit tracing through a processor reset. If the PTM is reset, tracing stops until the PTM is re-programmed and re-enabled. However, if the processor is reset, the last waypoint provided by the processor before the reset might not be traced.

1.6 Product documentation, design flow, and architecture

This section describes the content of the product books, how they relate to the design flow, and the relevant architectural standards and protocols.

Note

See *Additional reading* on page vi for more information about the books described in this section.

1.6.1 Documentation

The following books describe the PTM:

Technical Reference Manual

The *Technical Reference Manual* (TRM) describes the PTM functionality and the effects of functional options on the behavior of the PTM. It is required at all stages of the design flow. Some behavior described in the TRM might not be relevant, because of the way the PTM has been implemented and integrated. If you are programming the PTM, contact the implementer to determine the build configuration of the implementation, and the integrator to determine the pin configuration of the SoC that you are using.

Configuration and Sign-Off Guide

The *Configuration and Sign-Off Guide* (CSG) describes:

- the available build configuration options and related issues in selecting them
- how to configure the *Register Transfer Level* (RTL) with the build configuration options
- the processes to sign off the configured RTL and final macrocell.

The ARM product deliverables include reference scripts and information about using them to implement your design. Reference methodology documentation from your EDA tools vendor complements the CSG. The CSG is a confidential book that is only available to licensees.

Integration Manual

The *Integration Manual* (IM) describes how to integrate the PTM into a SoC including describing the pins that the integrator must tie off to configure the macrocell for the required integration. Some of the integration is affected by the configuration options that were used to implement the PTM. Contact the implementer of the macrocell that you are using to determine the implemented build configuration options. The IM is a confidential book that is only available to licensees.

1.6.2 Design flow

The PTM is delivered as synthesizable RTL. Before it can be used in a product, it must go through the following process:

1. Implementation. The implementer configures and synthesizes the RTL to produce a hard macrocell.
2. Integration. The integrator integrates the hard macrocell into a SoC.
3. Programming. The debug software developer programs the device.

Each stage of the process:

- can be performed by a different party
- can include options that affect the behavior and features at the next stage:

  Build configuration

  The PTM has no implementer defined options.

  Configuration inputs

  The integrator configures some features of the PTM by tying inputs to specific values. These configurations affect the start-up behavior before any software configuration is made. They can also limit the options available to the software.

  Software configuration

  The programmer configures the PTM by programming particular values into software-visible registers. This affects the behavior of the PTM.

  See Chapter 2 *Programmers Model* for information on the PTM registers.

——— **Note** ———

This manual refers to *implementation-defined* features that are applicable to build configuration options. References to a feature that is *included*, mean that the appropriate build and pin configuration options have been selected, while references to an *enabled* feature, mean one that has also been configured by software.

———————————

1.6.3 Architecture information

The PTM complies with, or implements, the specifications described in:

- *Trace macrocell*
- *Advanced Microcontroller Bus Architecture*.

This TRM complements architecture reference manuals, architecture specifications, protocol specifications, and relevant external standards. It does not duplicate information from these sources.

Trace macrocell

The PTM implements the Program Flow Trace architecture version 1.0. See CoreSight Program Flow Trace Architecture Specification.

Advanced Microcontroller Bus Architecture

The PTM complies with the AMBA 3 protocol. See *AMBA 3 APB Protocol Specification* and *AMBA 3 ATB Protocol Specification*.

An architecture specification typically defines a number of versions, and includes features that are either optional or partially specified, that is, are implementation- defined. The TRM describes which architectures are used, including which version is implemented, and the architectural choices made for the implementation. The TRM does not provide detailed information about the architecture, but some architectural information is included to give an overview of the implementation or, in the case of control registers, to make the manual easier to use. See the appropriate specification for more information about the implemented architectural features.

1.7 Product revisions

This section describes the differences in functionality between product revisions:

r0p0	First release.
r0p0-r1p0	Processor PMU event bus size increased from 29 to 52.

Chapter 2
Programmers Model

This chapter describes the programmers model. It contains the following sections:

2.1 About this programmers model

This chapter describes the mechanisms for programming the registers used to set up the trace and triggering facilities of the macrocell. The programmers model enables you to use the PTM registers to control the macrocell.

The following information applies to the CoreSight PTM A9 registers:

- The base address is not fixed, and can be different for any particular system implementation. The offset of each register from the base address is fixed.
- Do not attempt to access reserved or unused address locations. Attempting to access these locations can result in Unpredictable behavior.
- Unless otherwise stated in the accompanying text:
  - do not modify undefined register bits
  - ignore undefined register bits on reads
  - all register bits are reset to a logic 0 by a system or power-on reset.
- Access type in Table 2-2 on page 2-7 is described as follows:

RW	Read and write.
RO	Read only.
WO	Write only.

2.2 Modes of operation

When the PTM is powered on or reset, you must program all PTM registers before you enable tracing. If you do not do so, the trace results are Unpredictable.

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

You access the registers of the PTM through the CoreSight Debug APB bus. The PTM implements the CoreSight lock access mechanism, and can distinguish between memory-mapped accesses from on-chip software and memory-mapped accesses from a debugger, for example by using the *CoreSight Debug Access Port* (DAP).

See the *CoreSight Program Flow Trace Architecture Specification* for more information about programming the PTM.

The following sections describe how you control ETM programming.

- *Using the Programming bit*
- *Programming registers* on page 2-4.

2.2.1 Using the Programming bit

Use the Programming bit in the Main Control Register (see *Main Control Register* on page 2-10) to disable all operations during programming.

When the Programming bit is set to 0 you must not write to registers other than the Main Control Register, because this can lead to Unpredictable behavior.

When setting the Programming bit, you must not change any other bits of the Main Control Register. You must only change the value of bits other than the Programming bit of the Control Register when bit [1] of the Status Register is set to 1. ARM recommends that you use a read-modify-write procedure when changing the Main Control Register.

When the Programming bit is set to 1:

- The FIFO is permitted to empty and then no more trace is produced.
- The counters, sequencer, and start/stop block are held in their current state.
- The external outputs are forced LOW.

Figure 2-1 on page 2-4 shows a flow diagram of the procedure to program the PTM registers.

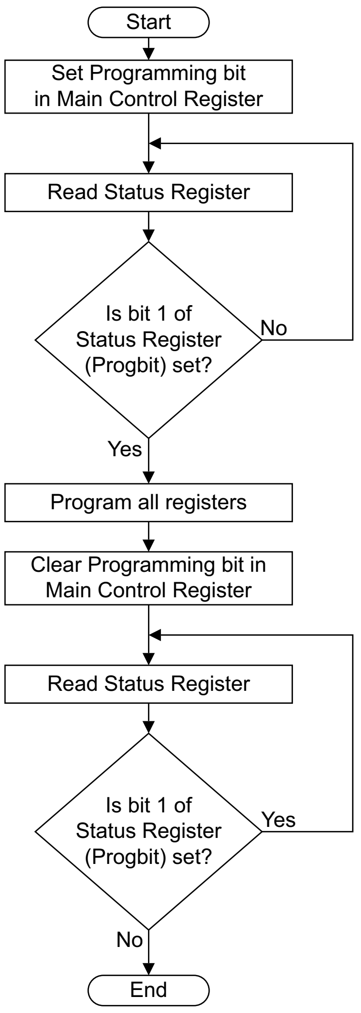
Start
Set Programming bit in Main Control Register
Read Status Register
Is bit 1 of Status Register (Progbit) set?
No
Yes
Program all registers
Clear Programming bit in Main Control Register
Read Status Register
Is bit 1 of Status Register (Progbit) set?
Yes
No
End

Figure 2-1 Programming the PTM registers

2.2.2 Programming registers

A reset of the PTM initializes the following registers:

- *Main Control Register* on page 2-10
- *Synchronization Frequency Register* on page 2-16
- *CoreSight Trace ID Register* on page 2-20
- the CoreSight registers at offsets `0xF00` - `0xFFC`
- *Peripheral Identification registers* on page 2-29
- *Component Identification registers* on page 2-29.

To start tracing, you must program the following registers to avoid Unpredicatble behavior:

- *Main Control Register* on page 2-10
- Trigger Event Register, TTER
- *TraceEnable Control registers* on page 2-13
- *CoreSight Trace ID Register* on page 2-20.

You might also have to program the following:

- *Address Comparator registers* on page 2-15

- *Counter registers* on page 2-16
- Sequencer registers if the sequencer is used
- External Output Event Registers if the external outputs are used
- Context ID comparator registers if the context ID comparator is used
- *Extended External Input Selection Register* on page 2-18 if the extended external inputs are used.

2.3 Register short names

All of the PTM registers have short names. Most of these are mnemonics for the full name of the register, except that the short name starts with the letters ETM, indicating that the register is defined by an ARM trace architecture. The ETM architecture is the original ARM trace architecture, and because register assignments are consistent across the trace architectures the register short names always take the ETM prefix. Table 2-1 gives some examples of the register short names.

Table 2-1 Examples of register short names

PTM register name	Register short name	Explanation of short name
Main Control Register	ETMCR	Trace Control Register
Trigger Event Register	ETMTRIGGER	Trace Trigger (Register)
Address Comparator Value Register 3	ETMACVR3	Trace Address Comparator Value Register 3

The use of the ETM prefix for the register short names means that the short names are distinct from the short names used for other registers, such as the processor control coprocessor registers and the debug registers.

2.4 Register summary

Table 2-2 shows the registers in offset order from the base memory address. For full descriptions of the PTM registers, see:

- *Register descriptions* on page 2-10, for the implementation-defined registers
- the *Program Flow Trace Architecture*, for the other registers.

Note

Registers not listed here are not implemented. Reading a non-implemented register address returns 0. Writing to a non-implemented register address has no effect.

All PTM registers are 32 bits wide.

The PTM registers are defined in the *Program Flow Trace Architecture Specification*. Table 2-2 lists all of the registers that are implemented in the PTM with their offsets from a base address. This base address is defined by the system integrator when placing the PTM in the Debug-APB memory map.

Table 2-2 PTM Register summary

Offset	Name	Type	Reset	Description
`0x0000`	ETMCR	RW	`0x401`	*Main Control Register* on page 2-10
`0x0004`	ETMCCR	RO	`0x8D294004`	*Configuration Code Register* on page 2-11
`0x0008`	ETMTRIGGER	RW	-	See *Program Flow Trace Architecture Specification*
`0x0010`	ETMSR	RW	-	*Status Register* on page 2-12
`0x0014`	ETMSCR	RO	-	*System Configuration Register* on page 2-13
`0x0018`	ETMTSSCR	RW	-	*TraceEnable Control registers* on page 2-13
`0x0020`	ETMTEEVR	RW	-	See *Program Flow Trace Architecture Specification*
`0x0024`	ETMTECR1	RW	-	*TraceEnable Control Register 1* on page 2-14
`0x0040-0x005C`	ETMACVR1 - ETMACVR8	RW	-	See *Program Flow Trace Architecture Specification*
`0x0080-0x009C`	ETMACTR1 - ETMACTR8	RW	-	
`0x0140-0x0144`	ETMCNTRLDVR1 - ETMCNTRLDVR2	RW	-	
`0x0150-0x0154`	ETMCNTENR1 - ETMCNTENR2	RW	-	
`0x0160-0x0164`	ETMCNTRLDEVR1 - ETMCNTRLDEVR2	RW	-	
`0x0170-0x0174`	ETMCNTVR1 - ETMCNTVR2	RW	-	
`0x0180-0x0194`	ETMSQmnEVR (Event 1 - 6)	RW	-	
`0x019C`	ETMSQR	RW	-	
`0x01A0-0x01A4`	ETMEXTOUTEVR1 - ETMEXTOUTEVR2	RW	-	

Table 2-2 PTM Register summary (continued)

Offset	Name	Type	Reset	Description
`0x1B0`	ETMCIDCVR1	RW	-	See *Program Flow Trace Architecture Specification*
`0x1BC`	ETMCIDCMR	RW	-	
`0x1E0`	ETMSYNCFR	RW	`0x400`	*Synchronization Frequency Register* on page 2-16
`0x1E4`	ETMIDR	RO	`0x411CF301`	*ID Register* on page 2-16
`0x1E8`	ETMCCER	RO	`0x00C019A2`	*Configuration Code Extension Register* on page 2-17
`0x1EC`	ETMEXTINSELR	RW	-	*Extended External Input Selection Register* on page 2-18
`0x1F8`	ETMTSEVR	RW	-	See *Program Flow Trace Architecture Specification*
`0x1FC`	ETMAUXCR	RW	`0x0`	*Auxiliary Control Register* on page 2-19
`0x200`	ETMTRACEIDR	RW	`0x0`	*CoreSight Trace ID Register* on page 2-20
`0x304`	OSLSR	RO	`0x0`	*OS Lock Status Register* on page 2-21
`0x314`	ETMPDSR	RO	`0x1`	*Device Power-Down Status Register* on page 2-21
`0xEDC`	ITMISCOUT	WO	-	*Miscellaneous Outputs Register* on page 2-23
`0xEE0`	ITMISCIN	RO	-	*Miscellaneous Inputs Register* on page 2-24
`0xEE8`	ITTRIGGER	WO	-	*Trigger Register* on page 2-25
`0xEEC`	ITATBDATA0	WO	-	*ATB Data 0 Register* on page 2-25
`0xEF0`	ITATBCTR2	RO	-	*ATB Control 2 Register* on page 2-26
`0xEF4`	ITATBID	WO	-	*ATB Identification Register* on page 2-26
`0xEF8`	ITATBCTR0	WO	-	*ATB Control 0 Register* on page 2-27
`0xF00`	ETMITCTRL	RW	`0x0`	*Integration Mode Control Register* on page 2-28
`0xFB8`	ETMAUTHSTATUS	RO	-	See *Program Flow Trace Architecture Specification*
`0xFC8`	ETMDEVID	RO	`0x0`	
`0xFCC`	ETMDEVTYPE	RO	`0x13`	
`0xFD0`	Peripheral ID4	RO	`0x04`	See *Peripheral Identification registers* on page 2-29
`0xFD4`	Peripheral ID5	RO		
`0xFD8`	Peripheral ID6	RO		
`0xFDC`	Peripheral ID7	RO		
`0xFE0`	Peripheral ID0	RO	`0x50`	
`0xFE4`	Peripheral ID1	RO	`0xB9`	
`0xFE8`	Peripheral ID2	RO	`0x1B`	
`0xFEC`	Peripheral ID3	RO	`0x00`	

Table 2-2 PTM Register summary (continued)

Offset	Name	Type	Reset	Description
`0xFF0`	Component ID0	RO	`0x0D`	See *Component Identification registers* on page 2-29
`0xFF4`	Component ID1	RO	`0x90`	
`0xFF8`	Component ID2	RO	`0x05`	
`0xFFC`	Component ID3	RO	`0xB1`	

For more information about these registers and the packets implemented by the PTM, see the *Program Flow Trace Architecture Specification*.

2.5 Register descriptions

This section describes the CoreSight PTM A9 registers.Table 2-2 on page 2-7 provides cross references to individual registers.

2.5.1 Main Control Register

The ETMCR Register characteristics are:

Purpose	Controls general operation of the PTM, such as whether tracing is enabled.
Usage constraints	See *Using the Programming bit* on page 2-3 for usage constraints.
Configurations	Available in all configurations.
Attributes	See Table 2-2 on page 2-7.

Figure 2-2 shows the bit assignments.

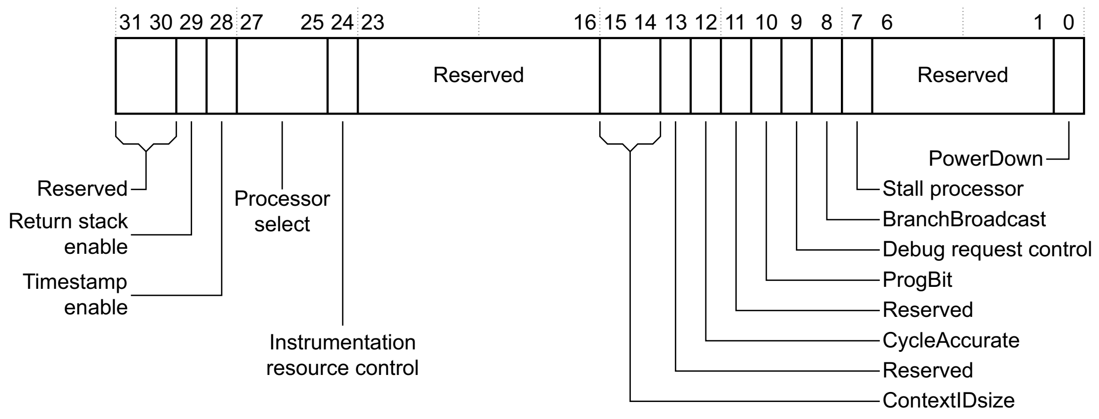

Figure 2-2 ETMCR Register bit assignments

Table 2-3 shows the bit assignments.

Table 2-3 ETMCR Register bit assignments

Bits	Name	Function
[31:30]	Reserved	SBZP

[29]	Return stack enable	Sets the Return stack enable:
		b0 = disabled
		b1 = enabled
		On reset, this bit is set to b0, disabled

[28]	Timestamp enable	Sets Timestamp enable:
		b0 = disabled
		b1 = enabled
		On reset, this bit is set to b0, disabled

[27:25]	Processor select	Select for external multiplexor if PTM is shared between multiple processors. See *Program Flow Trace Architecture Specification*.
[24]	Instrumentation Resource Control	RAZ, not implemented.
[23:16]	Reserved	SBZP

Table 2-3 ETMCR Register bit assignments (continued)

Bits	Name	Function

[15:14]	ContextIDSize	Sets the context ID size:
		b00 = no context ID tracing
		b01 = context ID bits [7:0] traced
		b10 = context ID bits [15:0] traced
		b11 = context ID bits [31:0] traced.
		On reset, this bit is set to b00, no context ID tracing

[13]	Reserved	SBZP

[12]	CycleAccurate	Sets cycle counting:
		b0 = cycle counting disabled
		b1 = cycle counting enabled
		On reset this bit is set to b0, no cycle counting.

[11]	Reserved	SBZP
[10]	Programming Bit	This bit must be set to b1 when the PTM is being programmed, see *Modes of operation* on page 2-3. On a PTM reset this bit is set to b1.
[9]	Debug request control	When set to b1 and the trigger event occurs, the **PTMDBGRQ** output is asserted until **PTMDBGACK** is observed. This enables a debugger to force the processor into Debug state. On PTM reset this bit is set to b0.
[8]	Branch Output	When this bit is set to b1, addresses are output for all executed branches, both direct and indirect. On PTM reset this bit is set to b0.
[7]	Stall Processor	RAZ - Not Implemented
[6:1]	Reserved	SBZP
[0]	PowerDown	This bit enables external control of the PTM. This bit must be cleared by the trace software tools at the beginning of a debug session. When this bit is set to b0, both the PTM and the trace interface in the processor are enabled. To avoid corruption of trace data, this bit must not be set before the Programming Status bit in the PTM Status Register has been read as 1. On PTM reset this bit is set to b1.

2.5.2 Configuration Code Register

The ETMCCR Register characteristics are:

Purpose	Provides information about the configuration of the PTM.
Usage constraints	There are no usage constraints.
Configurations	Available in all configurations.
Attributes	See Table 2-2 on page 2-7.

Figure 2-3 on page 2-12 shows the bit assignments.

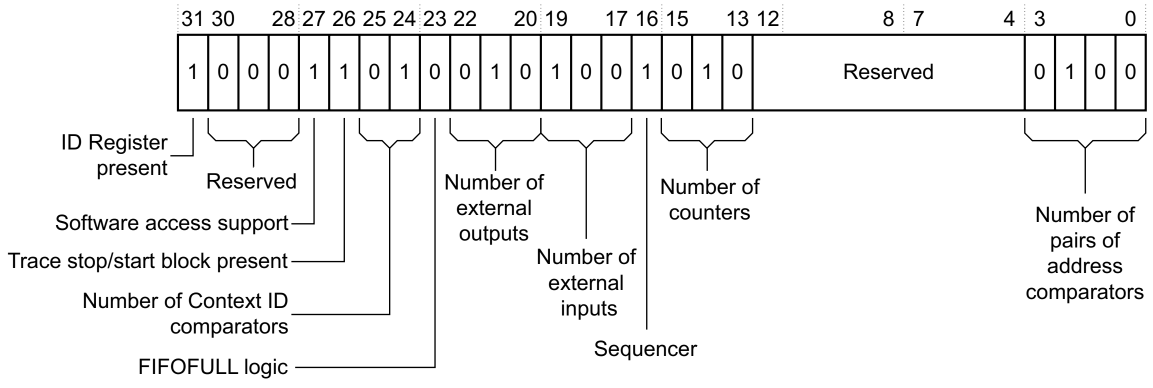

Figure 2-3 ETMCCR Register bit assignments

Table 2-4 shows the bit assignments.

Table 2-4 ETMCCR Register bit assignments

Bits	Name	Function
[31]	ID Register present	Indicates that the ID Register is present. See *ID Register* on page 2-16.
[30:28]	-	Reserved, RAZ on reads.
[27]	Software access	Indicates that software access is supported, see *Modes of operation* on page 2-3.
[26]	Trace stop/start block	Indicates that the trace start/stop block is present.
[25:24]	Number of Context ID comparators	Specifies the number of Context ID comparators, one.
[23]	FIFOFULL logic	Indicates that it is not possible to stall the processor to prevent FIFO overflow.
[22:20]	Number of external outputs	Specifies the number of external outputs, two.
[19:17]	Number of external inputs	Specifies the number of external inputs, four.
[16]	Sequencer	Indicates that the sequencer is present.
[15:13]	Number of counters	Specifies the number of counters, two.
[12:4]	-	Reserved
[3:0]	Number of pairs of address comparators	Specifies the number of address comparator pairs, four.

2.5.3 Status Register

The ETMSR Register characteristics are:

Purpose Provides information about the status of the trace and trigger logic. Bit [1] of this register shows the effective state of the Programming bit. You must wait for this bit to go to b1 before starting to program the PTM.

This register is described in the *CoreSight Program Trace Flow Architecture Specification*.

Usage constraints There are no usage constraints.

Configurations Available in all configurations.

Attributes See Table 2-2 on page 2-7.

2.5.4 System Configuration Register

The ETMSCR Register characteristics are:

Purpose	Shows the features supported by the ASIC. Some of the contents of this register are based on inputs provided by the ASIC.
Usage constraints	There are no usage constraints.
Configurations	Available in all configurations.
Attributes	See Table 2-2 on page 2-7.

Figure 2-4 shows the bit assignments.

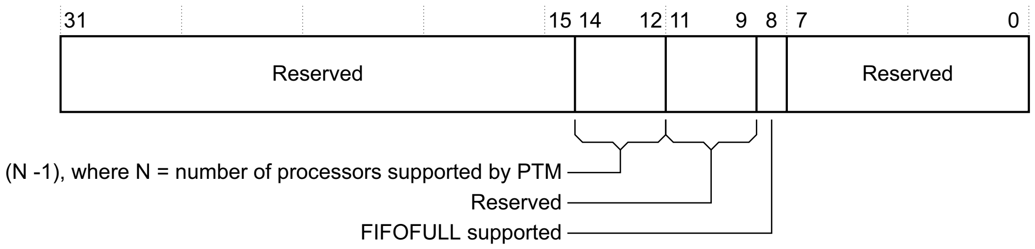

Figure 2-4 ETMSCR Register bit assignments

Table 2-5 shows the bit assignments.

Table 2-5 ETMSCR Register bit assignments

Bits	Function
[31:15]	Reserved, SBZP.
[14:12]	Number of supported processors minus 1. The value of this field is set by the **MAXCORES[2:0]** input to the PTM, see *Program Flow Trace Architecture Specification*.
[11:9]	Reserved, SBZP.
[8]	Read Only, as b0 - FIFOFULL is not supported.
[7:0]	Reserved, SBZP.

2.5.5 TraceEnable Control registers

The *CoreSight Program Flow Trace Architecture Specification* describes the TraceEnable mechanism that controls when trace is generated. Three registers control and configure the operation of TraceEnable. They are described in the following sections:

- *TraceEnable Start/Stop Control Register*
- TraceEnable Event Register, See *Program Flow Trace Architecture Specification*
- *TraceEnable Control Register 1* on page 2-14.

TraceEnable Start/Stop Control Register

The ETMTSSCR Register characteristics are:

Purpose	Specifies the single address comparators that hold the trace start and stop addresses that control the TraceEnable Start/Stop block.

Usage constraints There are no usage constraints.

Configurations Available in all configurations.

Attributes See Table 2-2 on page 2-7.

Figure 2-5 shows the bit assignments.

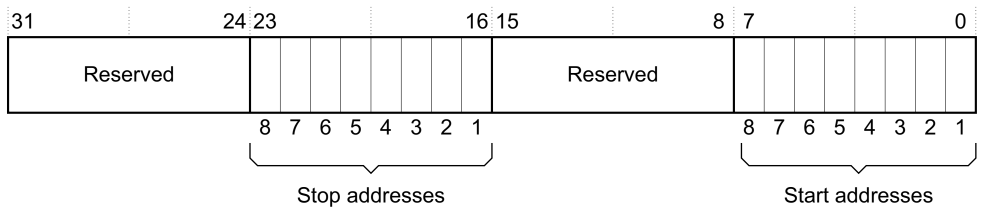

Figure 2-5 ETMSSCR Register bit assignments

Table 2-6 shows the ETMSSCR bit assignments.

Table 2-6 ETMSSCR Register bit assignments

Bits	Function
[31:24]	Reserved
[23:16]	When a bit is set to 1, it selects a single address comparator (8-1) as a stop address for the TraceEnable Start/Stop block. For example, if you set bit [16] to 1 it selects single address comparator 1 as a stop address.
[15:8]	Reserved
[7:0]	When a bit is set to 1, it selects a single address comparator (8-1) as a start address for the TraceEnable Start/Stop block. For example, if you set bit [0] to 1 it selects single address comparator 1 as a start address.

TraceEnable Control Register 1

The ETMTECR1 Register characteristics are:

Purpose The register:

- enables the start/stop logic
- specifies the address range comparators used for include or exclude control
- defines whether the specified address range comparators are used for include or exclude control.

Usage constraints There are no usage constraints.

Configurations Available in all configurations.

Attributes See Table 2-2 on page 2-7.

Table 2-5 on page 2-13 shows the bit assignments.

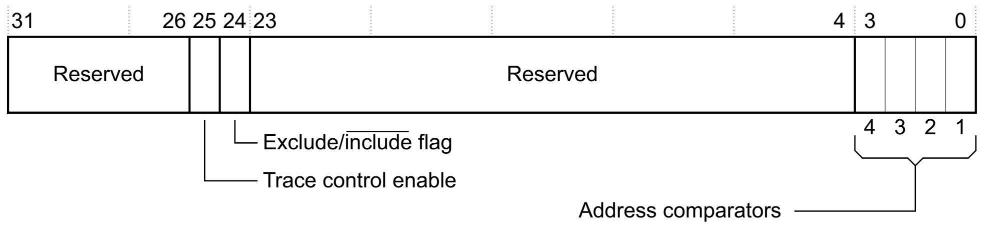

Figure 2-6 ETMTECT1 Register bit assignments

Table 2-7 shows the ETMTECR1 bit assignments.

Table 2-7 ETMTECR1 Register bit assignments

Bits	Function
[31:26]	Reserved, SBZP.
[25]	Trace start/stop control enable. The possible values of this bit are: **0** Tracing is unaffected by the trace start/stop logic. **1** Tracing is controlled by the trace on and off addresses configured for the trace start/stop logic. The trace start/stop resource, see Table 2-26 on page 2-30, is not affected by the value of this bit.
[24]	Exclude/include flag. The possible values of this bit are: **0** Include. The specified address range comparators indicate the regions where tracing can occur. No tracing occurs outside this region. **1** Exclude. The specified address range comparators indicate regions to be excluded from the trace. When outside an exclude region, tracing can occur.
[23:4]	Reserved, SBZP.
[3:0]	When a bit is set to 1, it selects an address range comparator, 4-1, for include/exclude control. For example, bit [0] set to 1 selects address range comparator 1.

Tracing all instructions

To trace all processor execution:

- set bit [24], the exclude/include flag, in this register to 1
- set all other bits in this register to 0
- set register `0x008`, the TraceEnable Event register, to `0x0000006F` (TRUE).

This has the effect of excluding nothing, that is, tracing everything, and setting the trace enable event to always true, with the start/stop logic ignored.

2.5.6 Address Comparator registers

The PTM implements eight pairs of Address Comparator registers. Table 2-8 shows the organization of these registers.

Table 2-8 PTM Address Comparator Registers

Address Comparator	Address Comparator Value Register	Address Comparator Access Type Register
1	ETMACVR1, register `0x010`, at offset `0x040`	ETMACTR1, register `0x020`, at offset `0x080`
2	ETMACVR2, register `0x011`, at offset `0x044`	ETMACTR2, register `0x021`, at offset `0x084`
3	ETMACVR3, register `0x012`, at offset `0x048`	ETMACTR3, register `0x022`, at offset `0x088`

Table 2-8 PTM Address Comparator Registers (continued)

Address Comparator	Address Comparator Value Register	Address Comparator Access Type Register
4	ETMACVR4, register `0x013`, at offset `0x04C`	ETMACTR4, register `0x023`, at offset `0x08C`
5	ETMACVR5, register `0x014`, at offset `0x050`	ETMACTR5, register `0x024`, at offset `0x090`
6	ETMACVR6, register `0x015`, at offset `0x054`	ETMACTR6, register `0x025`, at offset `0x094`
7	ETMACVR7, register `0x016`, at offset `0x058`	ETMACTR7, register `0x026`, at offset `0x098`
8	ETMACVR8, register `0x017`, at offset `0x05C`	ETMACTR8, register `0x027`, at offset `0x09C`

The *CoreSight Program Trace Flow Architecture Specification* describes the operation of these registers in the PTM.

2.5.7 Counter registers

The PTM implements two Counters, consisting of four registers for operation of each counter. Table 2-9 shows the organization of these registers.

Table 2-9 PTM Counter registers

Counter	Reload Value	Enable Event	Reload Event	Value
1	ETMCNTRLDVR1, register `0x050`, at offset `0x140`	ETMCNTENR1, register `0x054`, at offset `0x150`	ETMCNTRLDEVR1, register `0x058`, at offset `0x160`	ETMCNTVR1, register `0x05C`, at offset `0x170`
2	ETMCNTRLDVR2, register `0x051`, at offset `0x144`	ETMCNTENR2, register `0x055`, at offset `0x154`	ETMCNTRLDEVR2, register `0x059`, at offset `0x164`	ETMCNTVR2, register `0x05D`, at offset `0x174`

The *CoreSight Program Trace Flow Architecture Specification* describes the operation of these registers in the PTM.

2.5.8 Synchronization Frequency Register

The ETMSYNCFR Register characteristics are:

Purpose	Holds holds the trace synchronization frequency value.
Usage constraints	There are no usage constraints.
Configurations	Available in all configurations.
Attributes	See Table 2-2 on page 2-7.

Bits [2:0] of this register are not implemented and read as zero (RAZ). In all other respects, the *CoreSight Program Trace Flow Architecture Specification* describes the operation of this register in the PTM.

2.5.9 ID Register

The ETMIDR Register characteristics are:

Purpose	Holds the Program Flow Trace architecture variant, and defines the programmers model for the PTM.

Usage constraints There are no usage constraints.

Configurations Available in all CoreSight PTM configurations.

Attributes See Table 2-2 on page 2-7.

Figure 2-7 shows the ETMIDR bit assignments.

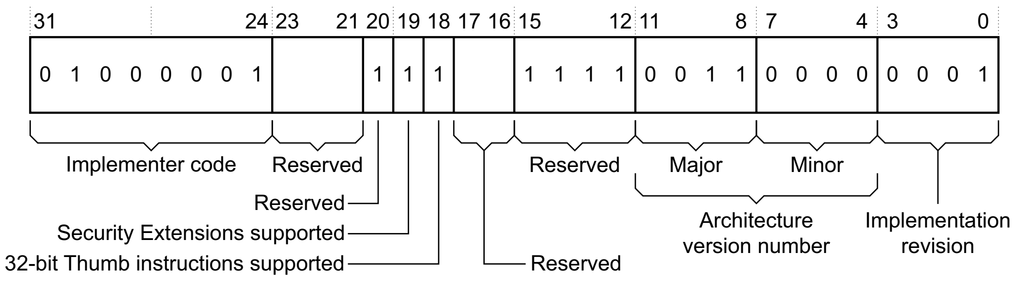

Figure 2-7 ETMIDR Register bit assignments

Table 2-10 shows the ETMIDR bit assignments. The values of the fields in this register are consistent with the ETM architecture, see *Embedded Trace Macrocell Architecture Specification*.

Table 2-10 ETMIDR Register bit assignments

Bits	Function
[31:24]	Implementer code. This field reads as `0x41`, ASCII code for A, indicating ARM Limited.
[23:21]	Reserved, RAZ.
[20]	Reserved. This bit is RAO.
[19]	Support for Security Extensions. The value of this bit is 1, indicating that the processor implements the ARM architecture Security Extensions.
[18]	Support for 32-bit Thumb instructions. The value of this bit is 1, indicating that a 32-bit Thumb instruction is traced as a single instruction.
[17:16]	Reserved, RAZ.
[15:12]	Reserved. This field reads as b1111.
[11:8]	b0011 - Major architecture version number.
[7:4]	b0000 - Minor architecture version number.
[3:0]	Implementation revision.

2.5.10 Configuration Code Extension Register

The ETMCCER Register characteristics are:

Purpose Provides additional information about the configuration of the PTM.

Usage constraints There are no usage constraints.

Configurations Available in all configurations.

Attributes See Table 2-2 on page 2-7.

Figure 2-8 on page 2-18 shows the bit assignments.

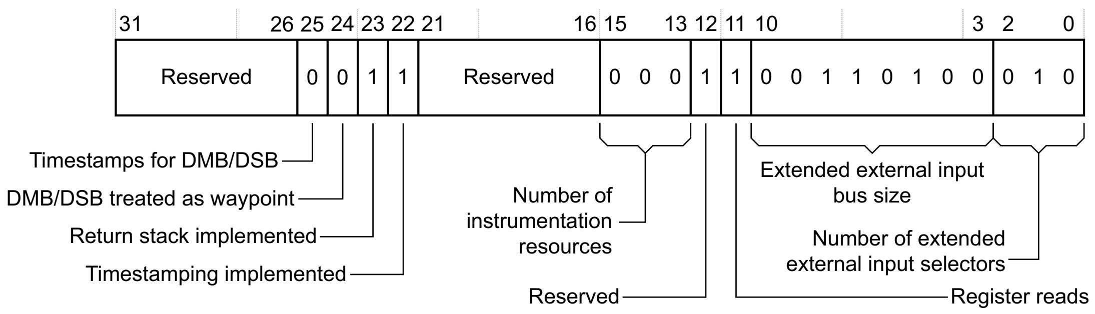

Figure 2-8 ETMCCER Register bit assignments

Table 2-11 shows the bit assignments.

Table 2-11 ETMCCER Register bit assignments

Bits	Function
[31:26]	Reserved, RAZ on reads.
[25]	b0 - Timestamps not generated for DMB/DSB.
[24]	b0 - DMB/DSB instructions are not treated as waypoints.
[23]	b1 - Return stack implemented.
[22]	b1 - Timestamping implemented.
[21:16]	Reserved, RAZ on reads.
[15:13]	b000 - Specifies the number of instrumentation resources.
[12]	Reserved, RAO.
[11]	b1 - Indicates that all registers, except some Integration Test Registers, are readable. See *Using the Integration Test Registers* on page 2-22 for details of the access permission to the Integration Test Registers. Registers with names that start with IT are the Integration Test Registers, for example ITATBCTR1.
[10:3]	b00110100 - Specifies the size of the extended external input bus, 52.
[2:0]	b010 - Specifies the number of extended external input selectors, 2.

2.5.11 Extended External Input Selection Register

The ETMEXTINSELR Register characteristics are:

Purpose	Selects inputs from the PMUEVENT bus to use as PTM resources.
Usage constraints	There are no usage constraints.
Configurations	Available in all configurations.
Attributes	See Table 2-2 on page 2-7.

Figure 2-9 on page 2-19 shows the bit assignments.

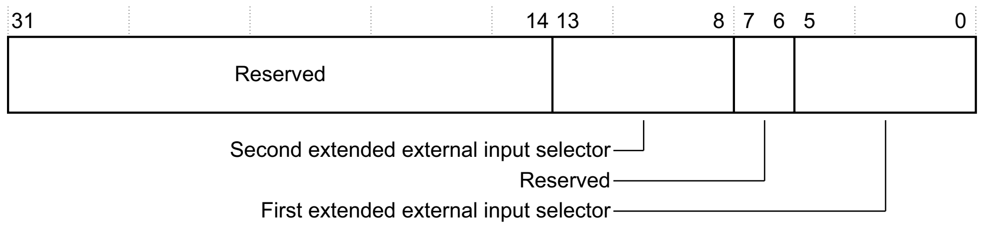
31
14 13
8 7 6 5
0
Reserved
Second extended external input selector
Reserved
First extended external input selector

Figure 2-9 ETMEXTINSELR Register bit assignments

Table 2-12 shows the bit assignments.

Table 2-12 ETMEXTINSELR Register bit assignments

Bits	Function
[31:14]	Reserved
[13:8]	Second extended external input selector
[7:6]	Reserved
[5:0]	First extended external input selector

2.5.12 Auxiliary Control Register

The ETMAUXCR Register characteristics are:

Purpose	Provides additional controls.
Usage constraints	There are no usage constraints.
Configurations	Available in all configurations.
Attributes	See Table 2-2 on page 2-7.

Figure 2-10 shows the bit assignments.

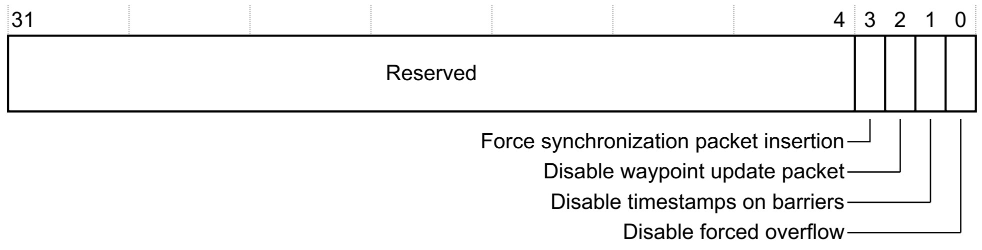
31
4 3 2 1 0
Reserved
Force synchronization packet insertion
Disable waypoint update packet
Disable timestamps on barriers
Disable forced overflow

Figure 2-10 ETMAUXCR Register bit assignments

Table 2-13 shows the bit assignments.

Table 2-13 ETMAUXCR Register bit assignments

Bits	Name	Function
[31:4]	-	Reserved
[3]	Force synchronization packet insertion	Force insertion of synchronization packets, regardless of trace activity. Possible values for this bit are: b0 = Synchronization packets delayed when trace activity is high. This is the reset value. b1 = Synchronization packets inserted regardless of trace activity. This bit might be set if synchronization packets occur too far apart. Setting this bit might cause the trace FIFO to overflow more frequently when trace activity is high.
[2]	Disable waypoint update packet	Specifies whether the PTM issues waypoint update packets if there are more than 4096 bytes between waypoints. Possible values for this bit are: b0 = PTM always issues update packets if there are more than 4096 bytes between waypoints. This is the reset value. b1 = PTM does not issue waypoint update packets unless required to do so as the result of an exception or debug entry.
[1]	Disable timestamps on barriers	Specifies whether the PTM issues a timestamp on a barrier instruction. Possible values for this bit are: b0 = PTM issues timestamps on barrier instructions. This is the reset value. b1 = PTM does not issue timestamps on barriers
[0]	Disable forced overflow	Specifies whether the PTM enters overflow state when synchronization is requested, and the previous synchronization sequence has not yet completed. This does not affect entry to overflow state when the FIFO becomes full. Possible values for this bit are: b0 = Forced overflow enabled. This is the reset value. b1 = Forced overflow disabled.

When setting any of bits [2:0] of this register the PTM behavior might contradict the Program Flow Trace Architecture Specification. Tools must be aware of the implications of setting any of these bits:

- Bit [0] might be set if the FIFO overflows because of the forced overflow condition. See the *Program Flow Trace Architecture Specification* for details on this condition. If this bit is set, tools must be aware that synchronization might not occur within the required synchronization period.
- Bit [1] might be set if timestamp packets are not required on barrier instructions. Typically, this might be set when using timestamping as a low-bandwidth measure of time, but might make correlation of multiple trace sources impossible.
- Bit [2] might be set if tools do not require the waypoint update packet that is output if there are more than 4096 bytes between waypoints.

2.5.13 CoreSight Trace ID Register

The ETMTRACEIDR Register characteristics are:

Purpose	Defines a 7-bit Trace ID for output to the trace bus. Before trace is generated, you must program this register with a non-reserved value. Reserved values are `0x00` and any value in the range `0x70-0x7F`.

Usage constraints	There are no usage constraints.
Configurations	Available in all configurations.
Attributes	See Table 2-2 on page 2-7.

See the *CoreSight Program Trace Flow Architecture Specification* for a description of the operation of this register in the PTM.

2.5.14 OS Lock Status Register

The OSLSR Register characteristics are:

Purpose	Shows whether trace register locking is implemented for the PTM macrocell.
Usage constraints	There are no usage constraints.
Configurations	Available in all configurations.
Attributes	See Table 2-2 on page 2-7.

For the PTM, the OSLSR *Reads As Zero* (RAZ) to show that OS Locking is not implemented.

2.5.15 Device Power-Down Status Register

The ETMPDSR Register characteristics are:

Purpose	Indicates the power-down status of the PTM.
Usage constraints	There are no usage constraints.
Configurations	Available in all configurations.
Attributes	See Table 2-2 on page 2-7.

This register always reads as `0x00000001`, indicating that the PTM Trace Registers can be accessed.

2.5.16 Integration registers

A CoreSight PTM can provide registers that let you test your integration of the PTM into your design. This section describes the integration registers that are implemented for the PTM. It contains the following sections:

- *Using the Integration Test Registers* on page 2-22
- *Miscellaneous Outputs Register* on page 2-23
- *Miscellaneous Inputs Register* on page 2-24
- *Trigger Register* on page 2-25
- *ATB Data 0 Register* on page 2-25
- *ATB Control 2 Register* on page 2-26
- *ATB Identification Register* on page 2-26
- *ATB Control 0 Register* on page 2-27.

Using the Integration Test Registers

Table 2-14 lists the Integration Test Registers implemented by the PTM.

Table 2-14 Integration Test Registers

Base offset	Function	Type	Description
`0xEDC`	Miscellaneous Outputs	WO	See *Miscellaneous Outputs Register* on page 2-23
`0xEE0`	Miscellaneous Inputs	RO	See *Miscellaneous Inputs Register* on page 2-24
`0xEE8`	Trigger	WO	See *Trigger Register* on page 2-25
`0xEEC`	ATB Data	WO	See *ATB Data 0 Register* on page 2-25
`0xEF0`	ATB Control 2	RO	See *ATB Control 2 Register* on page 2-26
`0xEF4`	ATB Identification	WO	See *ATB Identification Register* on page 2-26
`0xEF8`	ATB Control 0	WO	See *ATB Control 0 Register* on page 2-27

To access these registers you must first set bit [0] of the Integration Mode Control Register to 1. See *Integration Mode Control Register* on page 2-28. When this bit is set:

- You can use the write-only Integration Test Registers to set the outputs of some of the PTM signals. Table 2-15 lists the signals that you can control in this way.
- You can use the read-only Integration Test Registers to read the state of some of the PTM input signals. Table 2-16 on page 2-23 lists the signals that you can read in this way.

See the *Program Flow Trace Architecture Specification* for more information.

Table 2-15 Output signals that can be controlled by the Integration Test Registers

Register	Signal	Bits	Description
ITMISCOUT	**PTMEXTOUT[1:0]**	[9:8]	See *Miscellaneous Outputs Register* on page 2-23
	PTMIDLEnACK	[5]	
	PTMDBGREQ	[4]	
ITATBDATA0	**ATDATAM[31]**	[4]	See *ATB Data 0 Register* on page 2-25
	ATDATAM[23]	[3]	
	ATDATAM[15]	[2]	
	ATDATAM[7]	[1]	
	ATDATAM[0]	[0]	
ITTRIGGER	**PTMTRIGGER**	[0]	See *Trigger Register* on page 2-25
ITATBID	**ATIDM[6:0]**	[6:0]	See *ATB Identification Register* on page 2-26
ITATBCTR0	**ATBYTESM[9:8]**	[9:8]	See *ATB Control 0 Register* on page 2-27
	AFREADYM	[1]	
	ATVALIDM	[0]	

Table 2-16 shows the input signals that can be read by the integration test registers.

Table 2-16 Input signals that can be read by the Integration Test Registers

Register	Signal	Bits	Description
ITMISCIN	**STANDBYWFI**	[6]	See *Miscellaneous Inputs Register* on page 2-24
	PTMDBGACK	[4]	
	PTMEXTIN[3:0]	[3:0]	
ITATBCTR2	**AFVALIDM**	[1]	See *ATB Control 2 Register* on page 2-26
	ATREADYM	[0]	

The *CoreSight Components Technical Reference Manual* gives a full description of the use of the Integration Test Registers to check integration. In brief:

- When bit [0] of the Integration Mode Control Register is set, values written to the write-only Integration Test Registers map onto the specified outputs of the PTM. For example, writing `0x3` to **ITMISCOUT[9:8]** causes **EXTOUT[1:0]** to take the value `0x3`.
- When bit [0] 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 the PTM. For example, if you read **ITMISCIN[3:0]** you obtain the value of **PTMEXTIN**.

Miscellaneous Outputs Register

The ITMISCOUT Register characteristics are:

Purpose	Controls signal outputs when bit [0] of the Integration Mode Control Register is set.
Usage constraints	There are no usage constraints.
Configurations	Available in all configurations.
Attributes	See Table 2-2 on page 2-7.

Figure 2-11 shows the bit assignments

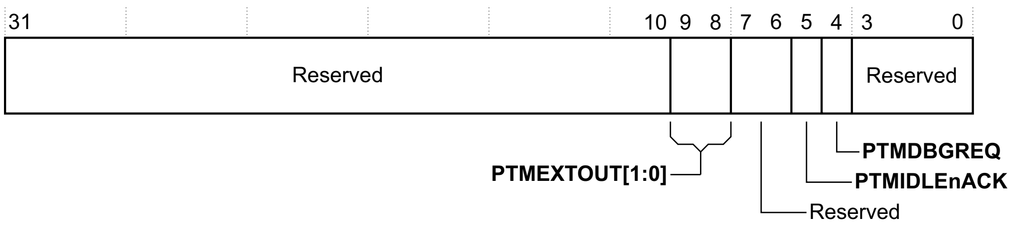

Figure 2-11 ITMISCOUT Register bit assignments

Table 2-17 shows the bit assignments.

Table 2-17 ITMISCOUT Register bit assignments

Bits	Function
[31:10]	Reserved, *Should Be Zero* (SBZ) on writes
[9:8]	Drives the **PTMEXTOUT[1:0]** outputs
[7:6}	Reserved, *Should Be Zero* (SBZ) on writes
[5]	Drives the **PTMIDLEnACK** output
[4]	Drives the **PTMDBGREQ** output
[3:0]	Reserved, *Should Be Zero* (SBZ) on writes

Miscellaneous Inputs Register

The ITMISCIN Register characteristics are:

Purpose	Enables the values of signal inputs to be read when bit [0] of the Integration Mode Control Register is set.
Usage constraints	There are no usage constraints.
Configurations	Available in all configurations.
Attributes	See Table 2-2 on page 2-7.

Figure 2-12 shows the bit assignments.

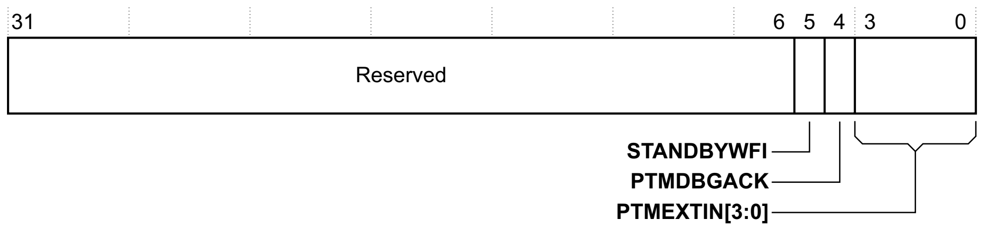

Figure 2-12 ITMISCIN Register bit assignments

Table 2-18 shows the bit assignments. The value of these fields depend on the signals on the input pins when the register is read.

Table 2-18 ITMISCIN Register bit assignments

Bits	Function
[31:6]	Reserved, RAZ on reads
[5]	Returns the value of the **STANDBYWFI** input
[4]	Returns the value of the **PTMDBGACK** input
[3:0]	Returns the value of the **PTMEXTIN[3:0]** inputs

Trigger Register

The ITTRIGGER Register characteristics are:

Purpose	Controls signal outputs when bit [0] of the Integration Mode Control Register is set.
Usage constraints	There are no usage constraints.
Configurations	Available in all configurations.
Attributes	See Table 2-2 on page 2-7.

Figure 2-13 shows the bit assignments.

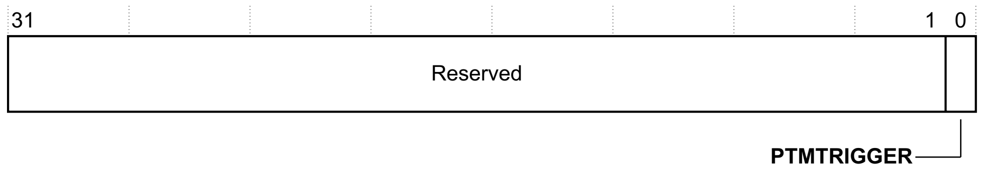

Figure 2-13 ITTRIGGER Register bit assignments

Table 2-19 shows the bit assignments.

Table 2-19 ITTRIGGER Register bit assignments

Bits	Function
[31:1]	Reserved, SBZ on writes
[0]	Drives the **PTMTRIGGER** output

ATB Data 0 Register

The ITATBDATA0 Register characteristics are:

Purpose	Controls signal outputs when bit [0] of the Integration Mode Control Register is set.
Usage constraints	There are no usage constraints.
Configurations	Available in all configurations.
Attributes	See Table 2-2 on page 2-7.

Figure 2-14 shows the bit assignments.

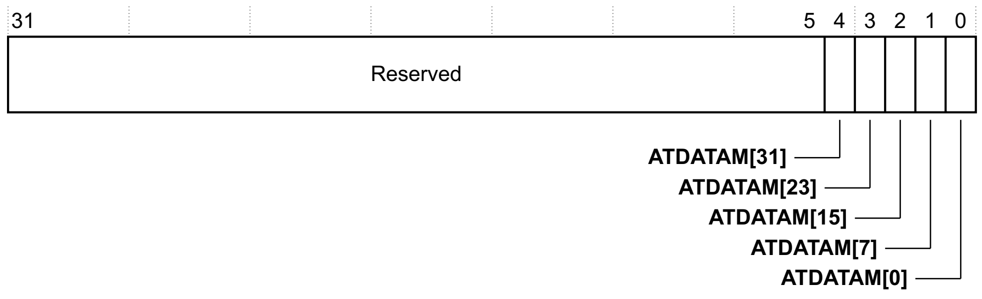

Figure 2-14 ITATBDATA0 Register bit assignments

Table 2-20 shows the bit assignments.

Table 2-20 ITATBDATA0 Register bit assignments

Bits	Function
[31:5]	Reserved, SBZ on writes
[4]	Drives the **ATDATAM[31]** output
[3]	Drives the **ATDATAM[23]** output
[2]	Drives the **ATDATAM[15]** output
[1]	Drives the **ATDATAM[7]** output
[0]	Drives the **ATDATAM[0]** output

ATB Control 2 Register

The ITATBCTR2 Register characteristics are:

Purpose	Enables the values of signal inputs to be read when bit [0] of the etmIntegration Mode Control Register is set
Usage constraints	There are no usage constraints.
Configurations	Available in all configurations.
Attributes	See Table 2-2 on page 2-7.

Figure 2-15 shows the bit assignments.

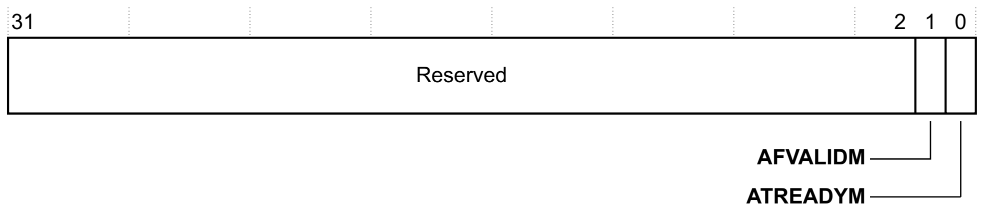

Figure 2-15 ITATBCT2 Register bit assignments

Table 2-21 shows the ITATBCTR2 bit assignments. The value of these fields depends on the signals on the input pins when the register is read.

Table 2-21 ITATBCTR2 Register bit assignments

Bits	Function
[31:2]	Reserved, RAZ on reads
[1]	Returns the value of the **AFVALIDM** input
[0]	Returns the value of the **ATREADYM** input

ATB Identification Register

The ITATBID Register characteristics are:

Purpose	Controls signal outputs when bit [0] of the Integration Mode Control Register is set.

Usage constraints There are no usage constraints.

Configurations Available in all configurations.

Attributes See Table 2-2 on page 2-7.

Figure 2-16 shows the bit assignments.

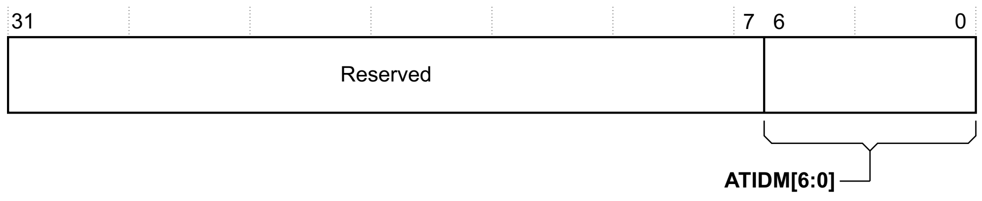

Figure 2-16 ITATBID Register bit assignments

Table 2-22 shows the ITATBID bit assignments.

Table 2-22 ITATBID Register bit assignments

Bits	Function
[31:7]	Reserved, SBZ on writes
[6:0]	Drives the **ATIDM[6:0]** outputs

ATB Control 0 Register

The ITATBCTR0 Register characteristics are:

Purpose Controls signal outputs when bit [0] of the Integration Mode Control Register is set.

Usage constraints There are no usage constraints.

Configurations Available in all configurations.

Attributes See Table 2-2 on page 2-7.

Figure 2-17 shows the ITATBCTR0 bit assignments.

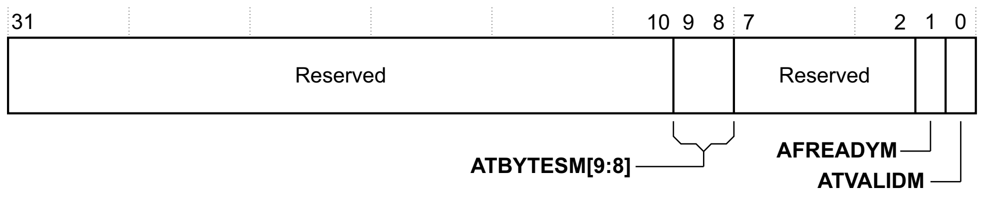

Figure 2-17 ITATBCTR0 Register bit assignments

Table 2-23 shows the ITATBCTR0 bit assignments.

Table 2-23 ITATBCTR0 Register bit assignments

Bits	Function
[31:10]	Reserved, SBZ on writes
[9:8]	Drives the **ATBYTESM** outputs
[7:2]	Reserved, SBZ on writes
[1]	Drives the **AFREADYM** output
[0]	Drives the **ATVALIDM** output

2.5.17 Integration Mode Control Register

The ETMITCTRL Register characteristics are:

Purpose	Enables topology detection and integration testing.
Usage constraints	Before entering integration mode, the PTM must be powered up and in programming mode. This means bit [0] of the Main Control Register is set to 0, and bit [10] of the Main Control Register is set to 1. After leaving integration mode, the PTM must be reset before attempting to perform tracing.
Configurations	Available in all configurations.
Attributes	See Table 2-2 on page 2-7.

Figure 2-18 shows the bit assignments.

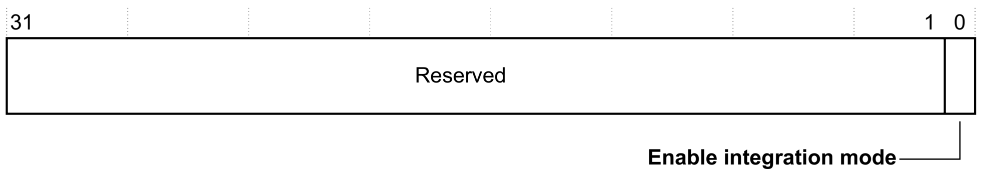

Figure 2-18 ETMITCTRL Register bit assignments

Table 2-24 shows the bit assignments.

Table 2-24 ETMITCTRL Register bit assignments

Bits	Function
[31:1]	Reserved.
[0]	When bit [0] is set to 1, the PTM enters an integration mode. On reset this bit is cleared to 0.

See the *CoreSight Program Trace Flow Architecture Specification* for a description of the operation of this register in the PTM.

2.5.18 Peripheral Identification registers

The peripheral identification registers provide standard information required for all CoreSight components. They are a set of eight registers, listed in register number order in Table 2-25.

Table 2-25 Summary of the peripheral identification registers

Register	Value	Offset
Peripheral ID4	`0x04`	`0xFD0`
Peripheral ID5	`0x00`	`0xFD4`
Peripheral ID6	`0x00`	`0xFD8`
Peripheral ID7	`0x00`	`0xFDC`
Peripheral ID0	`0x50`	`0xFE0`
Peripheral ID1	`0xB9`	`0xFE4`
Peripheral ID2[a]	`0x1B`	`0xFE8`
Peripheral ID3	`0x00`	`0xFEC`

a. Bits [7:4] of this value match the revision field in the ID Register, see *ID Register* on page 2-16.

Only bits [7:0] of each Peripheral ID Register are used, with bits [31:8] reserved. Together, the eight Peripheral ID Registers define a single 64-bit Peripheral ID

See the *CoreSight Program Trace Flow Architecture Specification* for a description of these registers.

2.5.19 Component Identification registers

There are four read-only Component Identification Registers, ComponentID0 to ComponentID3.

The component identification registers identify the PTM as a CoreSight component.

See the *CoreSight Program Trace Flow Architecture Specification* for a description of these registers.

2.6 Event definitions

As the *CoreSight Program Trace Flow Architecture Specification* describes, there are several event registers that can be programmed to select specific inputs as control events. Table 2-26 shows the event resources defined for the PTM.

Table 2-26 Event resource definitions

Resource type	Index values	Description
b000	0-7	Single address comparator 1-8
b001	0-3	Address range comparator 1-4
b100	0-1	Counter 1-2 at zero
b101	0-2	Sequencer in states 1-3
	8	Context ID comparator
	15	Trace start/stop resource
b110	0-3	External inputs 1-4
	8-9	Extended external input selectors 1-2
	13	Processor is in Non-secure state
	14	Trace prohibited by processor
	15	Hard-wired resource (always true)

2.7 Implementation-defined behavior

Exception return can always be deduced by examining the code image, so it is not necessary to explicitly flag these events. Exception return packets are only used when tracing ARMv7-M processors and therefore not generated by the PTM.

2.8 Turning off the PTM

During normal operation, the PTM buffers individual bytes of trace, and only generates output when at least 4 bytes are available. To enable the clocks to be gated, the PTM provides a mechanism to flush all trace out of the FIFOs in certain conditions. This differs from the AFVALID mechanism, that only ensures that trace up to a certain point has been output.

When the PTM has to enter an idle state, all trace in the FIFO is output. After the final data on the ATB interface has been accepted, the PTM is in an idle state.

Two conditions can cause the PTM to enter the idle state:

- the processor enters STANDBYWFI
- the Programming bit is set.

When the **STANDBYWFI** input is asserted, the PTM enters the idle state. When the PTM is in the idle state, and **STANDBYWFI** is still asserted, the PTM drives the **PTMIDLEnACK** output LOW. This can be used as an indication that the PTM is idle and in a safe quiescent state.

There is no requirement for the STANDBYWFI state to be maintained while the PTM is progressing towards the idle state, although the full sequence completes (with trace collecting in the FIFO) before the ATB output is resumed.

If the Programming bit is set, trace generation is stopped. The PTM then enters the idle state. This ensures that no trace packets remain. After the PTM completes the idle state transition with the Programming bit set, reading the Status Register reports the Programming bit as set. The Programming bit must not be cleared until the Status Register reports the Programming bit as set.

If the PTM is in the idle state as a result of STANDBYWFI and the Programming bit is set, the PTM ensures that any trace that is still in the main FIFO is drained before it updates the status register. Clocks are not gated internal to the PTM when it is in the idle state, unless the POWERDOWN bit is also set.

2.9 Interaction with the performance monitoring unit

The processor includes a *Performance Monitoring Unit* (PMU) that enables events, such as cache misses and instructions executed, to be counted over a period of time. This section describes how the PMU and PTM are used together.

2.9.1 Use of PMU events by the PTM

All PMU events except cycle count are available to the PTM through the extended external input facility. For more information on PMU events, see the *Cortex-A9 Technical Reference Manual*.

The PTM uses two extended external input selectors to access the PMU events. Each selector can independently select one of the PMU events, that is then active for the cycles in which the relevant events occur. These selected events can then be accessed by any of the event registers within the PTM.

Appendix A
Signal Descriptions

This appendix describes the PTM signals. It contains the following section:

A.1 PTM signal descriptions

This section describes the following signals:

- *Clocks and resets*
- *ASIC level signals*
- *Other signals* on page A-3
- *APB interface signals* on page A-3
- *ATB interface signals* on page A-4
- *Waypoint signals* on page A-4.

A.1.1 Clocks and resets

Table A-1 lists the clocks and resets.

Table A-1 Clocks and resets

Name	Description	Type
CLK	Global processor domain clock	Input
nPTMRESET	Reset for the **CLK** domain	Input
nCPURESET	Reset in the **CLK** domain for processor registers	Input
ATCLK	Clock for trace output	Input
ATCLKEN	Clock enable for **ATCLK**	Input
ATRESETn	Reset for the ATB domain	Input

A.1.2 ASIC level signals

Table A-2 lists the ASIC level signals.

Table A-2 ASIC level signals

Name	Description	Type	Clock domain
PTMPWRUP	PTM is active	Output	CLK
NIDEN	One of **NIDEN** and **DBGEN** must be HIGH for the PTM to be active	Input	Asynchronous
DBGEN		Input	Asynchronous
SE	Scan enable for test	Input	Asynchronous
PMUEVENT[51:0]	Processor PMU event bus	Input	CLK
TSVALUE[47:0]	Global system timestamp	Input	CLK
SYNCREQ	Global system synchronization request	Input	ATCLK
CLKCHANGE	Frequency change indication for **CLK** input	Input	CLK

A.1.3 Other signals

Table A-3 lists other signals.

Table A-3 Other signals

Name	Description	Type	Clock domain
PTMEXTIN[3:0]	External input resources.	Input	CLK
PTMEXTOUT[1:0]	External output resources.	Output	CLK
PTMTRIGGER	Trigger occurred status signal to CTI.	Output	CLK
STANDBYWFI	Processor is in WFI state and does not issue new waypoints. PTM can enter idle state.	Input	CLK
PTMIDLEnACK	PTM idle state indicator.	Output	CLK
PTMDBGREQ	Processor debug request signal.	Output	CLK
PTMDBGACK	Debug state indicator.	Input	CLK
MAXCORES[2:0]	Number of processors available for multiplexing this PTM to.	Input	CLK
CORESELECT[2:0]	Selected processor for multiplexing logic. Driven by bits [27:25] in the trace control register.	Output	CLK

A.1.4 APB interface signals

Table A-4 lists the APB interface signals.

Table A-4 APB interface signals

Name	Description	Type	Clock domain
PENABLEDBG	Data valid on **PRDATA** or **PWDATA**.	Input	CLK
PSELDBG	Selects the PTM APB slave interface.	Input	CLK
PADDRDBG[11:2]	APB address bus.	Input	CLK
PADDRDBG31	APB address bus.	Input	CLK
PWRITEDBG	APB transfer direction. HIGH for write, LOW for read.	Input	CLK
PWDATADBG[31:0]	APB write data bus.	Input	CLK
PRDATADBG[31:0]	APB read data bus.	Output	CLK
PREADYDBG	APB ready signal.	Output	CLK
PSLVERRDBG	APB error signal, always LOW.	Output	CLK

A.1.5 ATB interface signals

Table A-5 lists the ATB interface signals.

Table A-5 ATB interface signals

Name	Description	Type	Clock domain
ATDATAM[31:0]	32-bit trace data.	Output	ATCLK
ATIDM[6:0]	Trace source ID.	Output	ATCLK
ATVALIDM	**ATDATAM** is valid.	Output	ATCLK
ATBYTESM[1:0]	Number of valid bytes on **ATDATAM**.	Output	ATCLK
ATREADYM	Indicates that the trace port is able to accept the data on **ATDATAM**.	Input	ATCLK
AFVALIDM	The FIFO must be flushed.	Input	ATCLK
AFREADYM	Acknowledge for **AFVALIDM**. All trace generated when **AFVALIDM** was asserted has been output.	Output	ATCLK

A.1.6 Waypoint signals

Table A-6 lists the waypoint signals.

Table A-6 Waypoint signals

Signal	Description	Type	Clock domain
WPTTARGETPC[31:0]	Waypoint target address.	Input	CLK
WPTPC[31:0]	Waypoint last executed address indicator. This is the base LR in the case of an exception.	Input	CLK
WPTT32LINK	Indicates size of last executed address when in thumb state. Only valid when the waypoint is a branch and link.	Input	CLK
WPTCONTEXTID[31:0]	Context ID for instructions following this waypoint.	Input	CLK
WPTTRACEPROHIBITED	Trace is prohibited for current waypoint target.	Input	CLK
WPTnSECURE	Instructions following the current waypoint are executed in non-secure state.	Input	CLK
WPTTARGETJBIT	J bit for waypoint destination.	Input	CLK
WPTTARGETTBIT	T bit for waypoint destination.	Input	CLK
WPTEXCEPTIONTYPE[3:0]	Exception type.	Input	CLK
WPTLINK	Waypoint is a branch and updates the link register.	Input	CLK
WPTTYPE[2:0]	Waypoint type.	Input	CLK
WPTTAKEN	Waypoint passed its condition codes. Address is still used, irrespective of the value of this signal.	Input	CLK
WPTVALID	Waypoint is confirmed as valid.	Input	CLK

Table A-6 Waypoint signals (continued)

Signal	Description	Type	Clock domain
WPTCOMMIT[1:0]	Number of waypoints committed this cycle. It is valid to indicate a valid waypoint and commit it on the same cycle.	Input	CLK
WPTFLUSH	Flush signal from processor exception FIFO. All not yet committed waypoints are flushed.	Input	CLK
WPTFIFOEMPTY	Count of outstanding valid waypoints is zero, used for synchronization when the PTM is first enabled.	Input	CLK

Appendix B
Revisions

This appendix describes the technical changes between released issues of this book.

Table B-1 Issue A

Change	Location	Affects
First release	-	-

Table B-2 Differences between issue A and issue B

Change	Location	Affects
References to PTM-A9 changed to PTM	Throughout book	All revisions
PMUEVENT inputs changed from 29 to 52	Table 1-1 on page 1-5	r1p0
Deleted reference to level shifters	Figure 1-1 on page 1-4	All revisions
PMUEVENT size changed from [28:0] to [51:0]	Figure 1-3 on page 1-7 and Table A-2 on page A-2	r1p0
PTMPWRUP changed to **WPTENABLE**	Figure 1-3 on page 1-7	All revisions
ID Register bit [0] value changed from 0 to 1	Figure 2-7 on page 2-17	r1p0
Configuration Code Extension Register bits [10:3] revised to show increased size of **PMUEVENT**	Figure 2-8 on page 2-18 and Table 2-11 on page 2-18	r1p0
Extended External Input Selector Register bits revised	Figure 2-9 on page 2-19 and Table 2-12 on page 2-19	r1p0
Peripheral ID2 and Peripheral ID3 values revised	Table 2-25 on page 2-29	r1p0

Table B-3 Differences between issue B and issue C

Change	Location	Affects
ARM publications list updated	*ARM publications* on page vi	All revisions
Programmers Model updated to reflect the latest template	Chapter 2 *Programmers Model*	r1p0
Miscellaneous Inputs Register		All revisions
• ITMISCIN Register bits assignments figure updated	Figure 2-12 on page 2-24	
• ITMISCIN Register bits assignments table updated	Table 2-18 on page 2-24	