Arm® CoreSight™ SDC-600 Secure Debug Channel

Revision: r0p2

Technical Reference Manual

arm

101130_0002_02_en

Arm® CoreSight™ SDC-600 Secure Debug Channel

Technical Reference Manual

Release Information

Document History

Issue	Date	Confidentiality	Change
0001-00	09 March 2018	Non-Confidential	First release for r0p1
0002-00	30 May 2018	Non-Confidential	First release for r0p2
0002-01	27 June 2018	Non-Confidential	Second release for r0p2
0002-02	30 November 2018	Non-Confidential	Third release for r0p2

LES-PRE-20349

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.

Unrestricted Access is an Arm internal classification.

Product Status

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

Web Address

http://www.arm.com

Contents

Arm® CoreSight™ SDC-600 Secure Debug Channel Technical Reference Manual

Preface

This preface introduces the *Arm® CoreSight™ SDC-600 Secure Debug Channel Technical Reference Manual*.

It contains the following:

About this book

This document gives reference information for the SDC-600 Secure Debug Channel.

Product revision status

The r*m*p*n* identifier indicates the revision status of the product described in this book, for example, r*1*p*2*, where:

r*m* Identifies the major revision of the product, for example, r1.

p*n* Identifies the minor revision or modification status of the product, for example, p2.

Intended audience

This book is written for system designers, system integrators, and programmers who are designing or programming a *System-on-Chip* (SoC) that uses the Arm® SDC-600 Secure Debug Channel product.

Using this book

This book is organized into the following chapters:

Chapter 1 Introduction
This chapter provides an overview of the Arm CoreSight SDC-600 Secure Debug Channel.

Chapter 2 Functional description
This chapter describes the functionality of the SDC-600.

Chapter 3 Programmer's model
This chapter provides a description of the address map and registers of the SDC-600.

Appendix A Signal descriptions
This chapter describes the signals of the SDC-600.

Appendix B Revisions
This appendix describes 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.

See the *Arm® Glossary* for more information.

Typographic conventions

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.

`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>`
Encloses replaceable terms for assembler syntax where they appear in code or code fragments. For example:

```
MRC p15, 0, <Rd>, <CRn>, <CRm>, <Opcode_2>
```

SMALL CAPITALS
Used in body text for a few terms that have specific technical meanings, that are defined in the *Arm® Glossary*. For example, IMPLEMENTATION DEFINED, IMPLEMENTATION SPECIFIC, UNKNOWN, and UNPREDICTABLE.

Timing diagrams

The following figure 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, so 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.

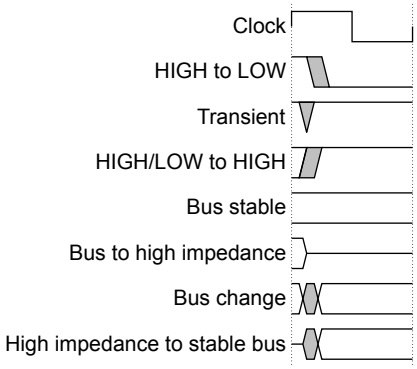

Figure 1 Key to timing diagram conventions

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.
- LOW for active-LOW signals.

Lowercase n

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

Additional reading

This book contains information that is specific to this product. See the following documents for other relevant information.

Arm publications

- *AMBA® APB Protocol Specification* (ARM IHI 0024C)
- *ARM® CoreSight™ Architecture Specification v3.0* (ARM IHI 0029E)
- *Arm® Debug Interface Architecture Specification ADIv5.0 to ADIv5.2* (ARM IHI 0031E)
- *Arm® Debug Interface Architecture Specification ADIv6.0* (Arm IHI 0074B)
- *ARM® AMBA® 5 AHB Protocol Specification* (ARM IHI 0033B.b)
- *AMBA® Low Power Interface Specification* (ARM IHI 0068C)
- *Advanced Communications Channel Architecture Specification* (ARM IHI 0076A)

Other publications

- *Verilog-2001 Standard (IEEE Std 1364-2001).*
- *Accellera, IP-XACT version 1685-2009.*

Feedback

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 *Arm CoreSight SDC-600 Secure Debug Channel Technical Reference Manual*.
- The number 101130_0002_02_en.
- If applicable, the page number(s) to which your comments refer.
- 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 provides an overview of the Arm CoreSight SDC-600 Secure Debug Channel.

It contains the following sections:

1.1 About

Arm CoreSight SDC-600 Secure Debug Channel provides a dedicated channel for authentication between an external debugger and a debug target platform by using an unlocking mechanism.

The SDC-600-based architecture provides an interface through which secure debug certificates can be injected to the platform. This is done in a standard way through the *Debug Access Port* (DAP), which is normally used to debug the platform. It eliminates the need for OEM proprietary delivery mechanisms for such certificates.

SDC-600 performs the following tasks:

- Requests power and optionally reboots the servicing agent.
- Establishes and maintains a link between a port on the external side, which is serviced by the debugger, and a port on the internal side, which is serviced by an agent on the target system.
- Transports messages from an external debugger to a hardware or software agent on a target system through a point-to-point link.

The debugged target and the servicing agent are typically the same processor or processor subsystem, but they can be separate entities as well.

The authentication process can involve a hardware- or software-based cryptographic engine on the target. The cryptographic engine verifies the debug certificate that is passed to the servicing agent through the SDC-600. The debugger and the servicing agent run a protocol on top of the SDC-600, which:

1. Identifies the SoC (SoC_ID).
2. Injects the appropriate debug certificate to the debug target for processing by the cryptographic engine.

The following is a high-level description of a sample authentication process:

1. The debugger wants to access the target's debug resources.
2. The debugger uses the CoreSight ID registers and discovery process to identify the SDC-600's external interface.
3. The debugger accesses the SDC-600 to start the unlocking process.
4. The SDC-600 requests the powerup of the rest of its functional blocks.
5. The debugger asks for a SoC_ID from the servicing target to identify the target system.
6. A certificate is generated by the debugger for the SoC_ID that is transmitted to the servicing target.
7. The servicing agent decides whether the debugger has the rights to access the debug target based on the provided certificate.
8. If access is granted, the target agent drives the authentication signals accordingly on the Access Ports so that the connected devices can be accessed by the debugger.

The following terminology is used throughout the document:

External
: The component or the end of the communication channel that is connected to the debugger through the Debug Port.

Internal
: The component or the end of the communication channel that is connected to the servicing agent.

Servicing agent
: The agent on the internal side that implements authentication by checking the certificate and controls the authentication signals in the target system. It communicates through the SDC-600 and services the interrupts that are generated by the internal COM Port component. The servicing agent can be implemented as software executing on the target processor, or on a separate processor in a secure island or subsystem.

Target
: The debug target that is requesting debug authentication. In some systems, the servicing agent can be implemented as code which runs on the target processor.

1.2 Compliance

SDC-600 is compatible with the following protocol specifications and standards:

- *ARM® CoreSight™ Architecture Specification v3.0* (ARM IHI 0029E).
- *AMBA® APB Protocol Specification* (ARM IHI 0024C).
- *ARM® AMBA® 5 AHB Protocol Specification* (ARM IHI 0033B.b).
- *Arm® Debug Interface Architecture Specification ADIv5.0 to ADIv5.2* (ARM IHI 0031E).
- *Arm® Debug Interface Architecture Specification ADIv6.0* (Arm IHI 0074B).
- *AMBA® Low Power Interface Specification* (ARM IHI 0068C).
- *Advanced Communications Channel Architecture Specification* (ARM IHI 0076A).
- *Accellera, IP-XACT version 1685-2009*.

1.3 Configurations

The SDC-600 Secure Debug Channel is designed to be integrated into three different types of systems.

These systems are:

- ADIv6-compliant systems, including implementations built using Arm CoreSight SoC-600.
- ADIv5.2-compliant systems, including implementations built using Arm CoreSight SoC-400.
- Systems that have processor cores with Integrated Arm Cortex®-M DAP.

The components and interfaces that you use for integration differ depending on your system.

The SDC-600 creates a communication channel between the debugger and the servicing agent. Any of the above system configurations can be further differentiated based on what the servicing agent is:

- Systems that have a dedicated servicing agent, such as an Arm CryptoIsland subsystem.
- Systems that run code on the target processor as all or part of the servicing agent. For example, this can be a pure software implementation, or even a driver implementation that uses an external Arm CryptoCell product for part of the processing work.

1.4 Components

The SDC-600 Secure Debug Channel can be integrated into various systems, and the components that are used vary depending on the system.

The SDC-600 contains the following components:

- An External component, which is used on the debugger side and recognized as a special access port (COM-AP) or a CoreSight peripheral device (APBCOM) by the debugger. The External component has three different variants depending on the debug infrastructure version:
  - The External APBCOM, which is used in ADIv6-compliant systems.
  - The External APBCOM for Integrated Cortex-M DAP, which is used with processor cores with Integrated Cortex-M DAP. This component is a CoreSight ROM table that includes COM functionality.
  - The COM-AP, which is used in ADIv5.2-compliant systems.
- An Internal component, which is the Internal APBCOM. The Internal component is accessible by the target processor core through the system interconnect.
- A COM asynchronous bridge, which transfers data between different clock and power domains. The asynchronous bridge has two variants:
  - An area-optimized bridge (Direct Bridge), which can be used in systems where the internal and external components are in neighboring power and clock domains.
  - A traditional bridge (Indirect Bridge), which can be used in any system, and can be instantiated multiple times on a CWI channel if passing multiple domains.
- A Debug Splitter, which is a module that is placed between the SLV master interface of the Integrated Cortex-M DAP and the SLV slave port of the processor core in non-AHB-Lite compliant mode. The Debug Splitter is only required with systems using the Integrated Cortex-M DAP of the processor core. The Debug Splitter supports the following Arm Cortex processor cores:
  - Arm Cortex-M0.
  - Arm Cortex-M0+.
  - Arm Cortex-M7.
  - Arm Cortex-M23.
  - Arm Cortex-M33.

Related concepts

2.1 About the functions on page 2-21

Related reference

2.3 Hardware components on page 2-24

1.5 Interfaces

The SDC-600 has the following external interfaces:

Table 1-1 SDC-600 interfaces

Interface name	Description
AMBA4 APB	Target slave interface on all programming interfaces (except for COM-AP) and register access infrastructure.
DAPBUS	Target slave interface on COM-AP programming interfaces and register access infrastructure. ——— **Note** ——— This interface is used instead of APB and **DP_ABORT** in ADIv5.2-compliant systems. ———
Cortex-M Debug	Cortex-M Debug interface can refer to a: • Target slave interface that is compatible with the debug master interface of the Integrated Cortex-M DAP of Cortex-M class processor cores. • Master interface that is compatible with the debug slave interface of the Cortex-M class processor cores in non-AHB mode.
LPI Q-Channel	Clock and power controlling interfaces intended for low-power usage.
COM Wire interface	Used for low-cost transport of bytes in a point-to-point system to enable communication from an external debugger to a hardware or software agent on a target system.
Powerup Request interface	Used to request power for the servicing agent.
Reboot Request interface	Used to send a request to reboot the servicing agent.
IRQ interface	Used to send an interrupt to the servicing agent.
DP_ABORT	An abort request causes the external side to terminate an ongoing transaction and free up its APB slave interface.

Related concepts

2.4 SDC-600 Interfaces on page 2-31

1.6 Design Process

The SDC-600 components are delivered as synthesizable Verilog RTL. Before the SDC-600 components can be used in a product, they must go through the following processes:

System design
Determining the necessary structure and interconnections of the SDC-600 components that form the CoreSight secure debug channel.

Configuration
Defining the memory map of the system and the functional configuration of the SDC-600 components.

Integration
Connecting the SDC-600 components together, and to the SoC memory and debug systems.

Verification
Verifying that the CoreSight secure debug channel has been correctly integrated to the processor or processors in your SoC.

Implementation
Using the Verilog RTL in an implementation flow to produce a hard macrocell.

The operation of the final device depends on:

Configuration
The implementer chooses the options that affect how the RTL source files are pre-processed. These options usually include, or exclude, logic that affects one or more of the area, maximum frequency, and features of the resulting macrocell.

Software configuration
The programmer configures the CoreSight secure debug channel by programming specific values into registers that affect the behavior of the SDC-600 components.

This section contains the following subsection:
- *1.6.1 Documentation* on page 1-17.

1.6.1 Documentation

Each Arm document has an intended audience and is associated with specific tasks in the design flow. These documents do not reproduce Arm architecture and protocol information.

The Arm SDC-600 documentation set is as follows:

Technical Reference Manual
The Technical Reference Manual (TRM) describes the functionality and the effects of functional options on the behavior of the SDC-600. It is required at all stages of the design flow. The choices that are made in the design flow can mean that some behaviors that are described in the TRM are not relevant. If you are programming the SDC-600, then contact the implementer to determine:
- The build configuration of the implementation.
- What integration, if any, was performed before implementing the SDC-600.

Configuration and Integration Manual
The Configuration and Integration Manual (CIM) describes:
- A list of the design-time configuration options.
- The available build configuration options and related issues in selecting them.
- How to configure the Register Transfer Level (RTL) model with the build configuration options.
- How to run test vectors.
- The processes to sign off the configured design.

Arm product deliverables include reference scripts and information about using them to implement your design.

The CIM is a confidential document that is only available to licensees.

1.7 Product revisions

This section describes the differences in functionality between product revisions of Arm SDC-600.

r0p1

First release of SDC-600.

r0p2

The following changes have been made in this release:

- The APB interfaces received a register stage to improve timing performance.
- The ports on the bridge internal interface have been renamed.
- The sdc600_debugsplitter signals have been changed.
- An additional Power LPI Q-Channel interface has been added to the external side of the sdc600_comasyncbridge_indirect. Therefore both bridge sides now have their own Power LPI interface.
- The Power LPI Q-Channel interface has been removed from the sdc600_comap, sdc600_apbcom_ext, and the sdc600_apbcom_ext_rom components. Power is secured through the powerup request interface of the debug infrastructure above in the hierarchy.
- The parameter `TIE_OFF_PRESENT` has been removed from sdc600_apbcom_ext_rom. Now, the revision and ID parameters must be set to valid values by the implementer.
- The parameter `ROM_TABLE_PTR` on sdc600_debugsplitter has been renamed to `ROM_ENTRY0` and extended to 32 bits to be aligned with other ROM tables.
- The **EXT_PWR_WAKE** signal has been added to both sides of sdc600_comasyncbridge_indirect.
- The **DP_ABORT**, **PSLVERR**, and **PWAKEUP** signals have been removed from the sdc600_apbcom_ext_rom component.
- The Internal APBCOM module now waits for the active clock to be confirmed on CLK LPI before accepting the PWR LPI state change request.

Chapter 2
Functional description

This chapter describes the functionality of the SDC-600.

It contains the following sections:

2.1 About the functions

This section describes the functional blocks in the SDC-600.

The functional blocks differ depending on the configuration.

The following figure shows the functional blocks when integrating the SDC-600 into ADIv6-compliant systems.

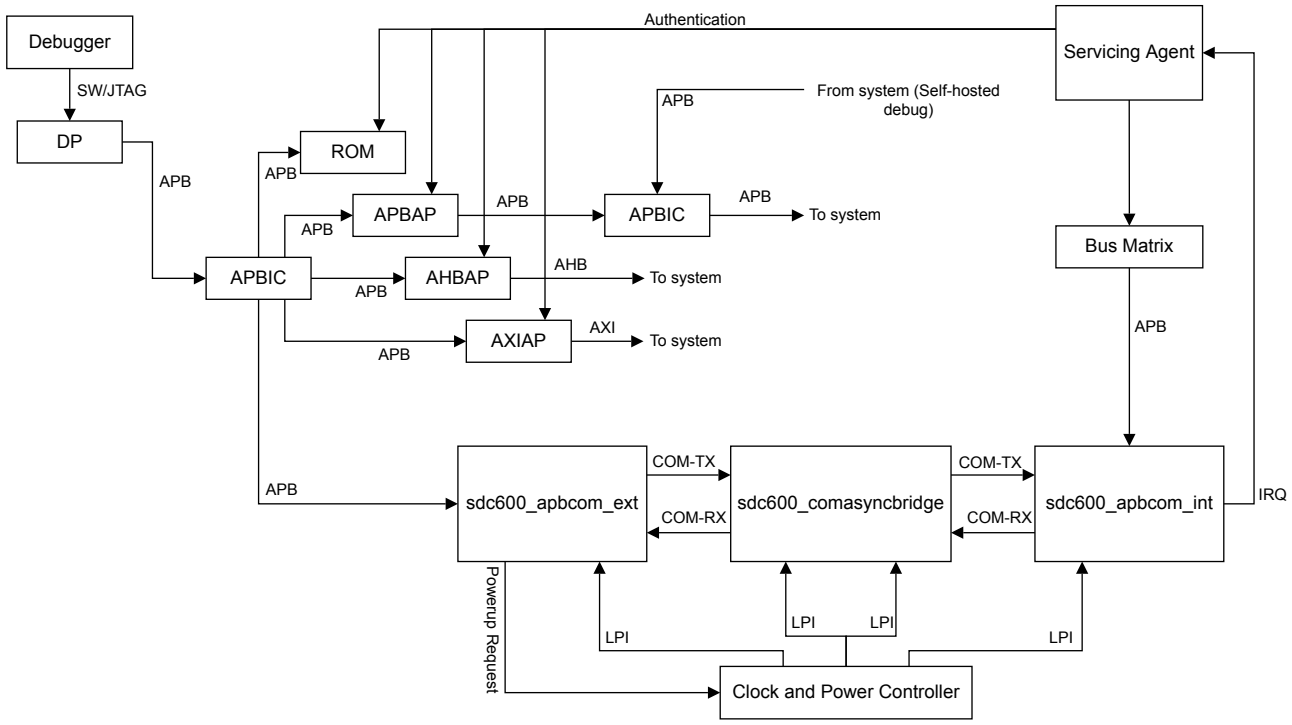

Figure 2-1 SDC-600 block diagram for ADIv6

The following figure shows the functional blocks when integrating the SDC-600 into ADIv5.2-compliant systems.

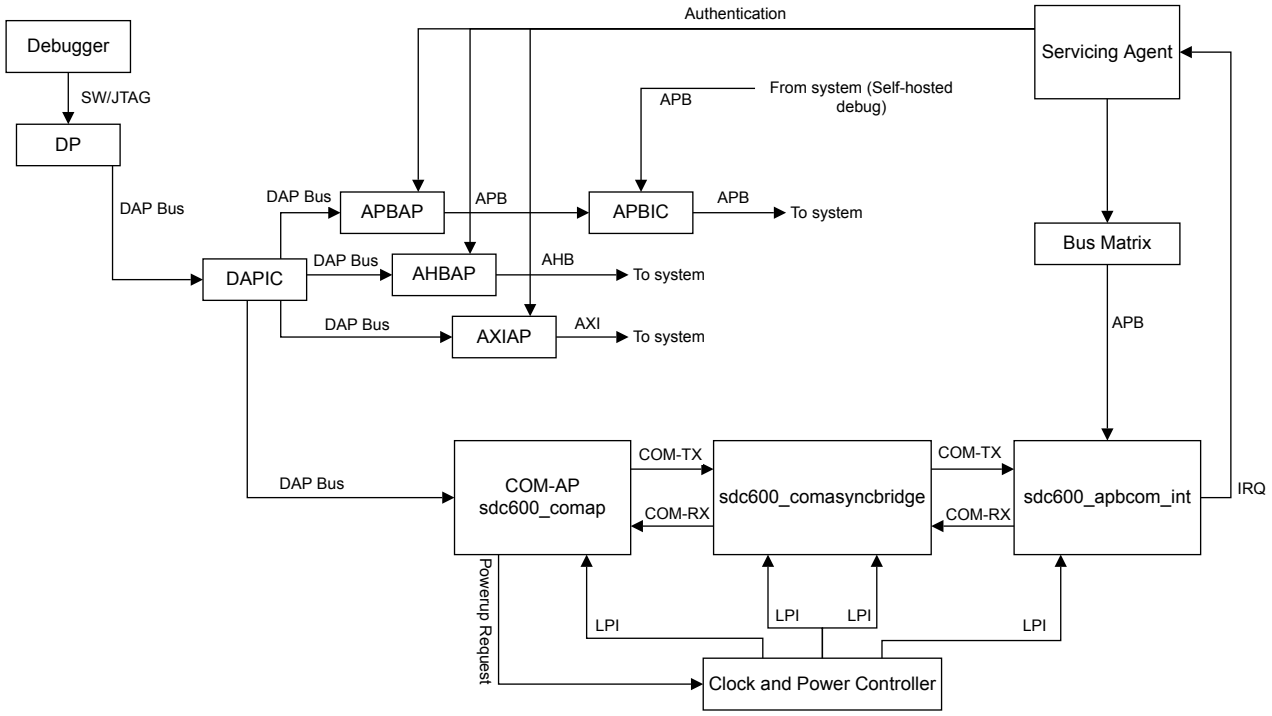

Figure 2-2 SDC-600 block diagram for ADIv5.2

The following figure shows the functional blocks when integrating the SDC-600 into systems with Integrated Cortex-M DAP.

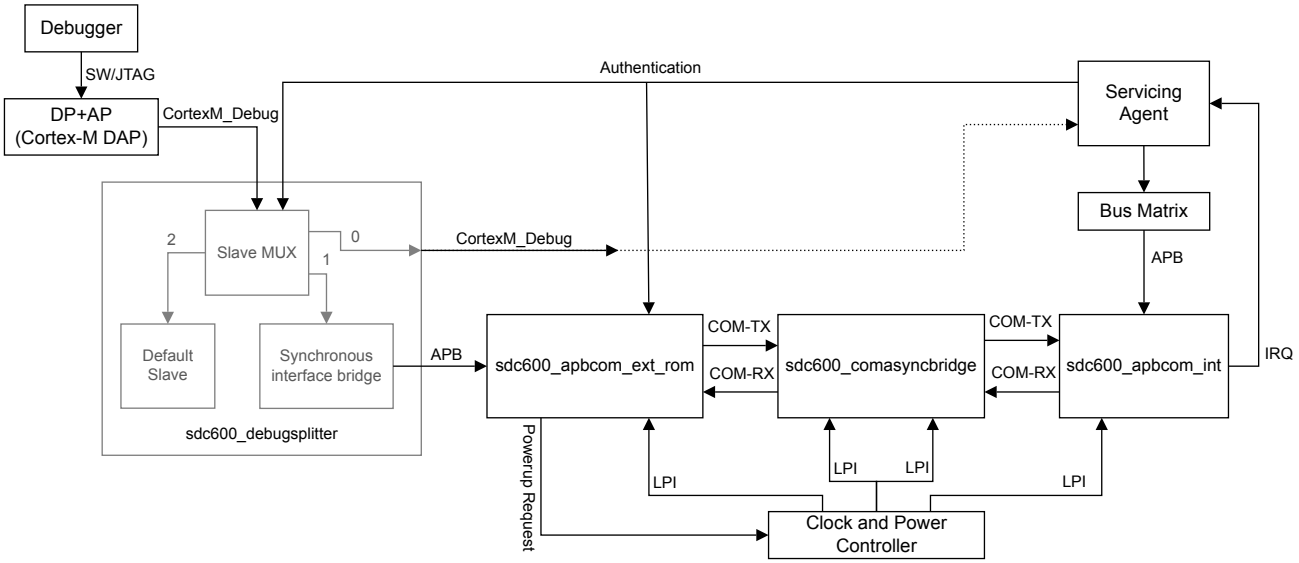

Figure 2-3 SDC-600 block diagram for Integrated Cortex-M DAP systems

2.2 Clocks

This section describes the clocks that are used in the SDC-600.

Table 2-1 Clocks used in SDC-600

Clock	Description
PCLK	Used by the: • External APBCOM. • External APBCOM for Integrated Cortex-M DAP. • Internal APBCOM. It is the clock of the related APB infrastructure.
DAPCLK	Used by the COM-AP module. It is the clock of the related DAP bus infrastructure.
CLK	Used by the Debug Splitter, and is also its interface clock for all of its AHB and APB buses. ——— **Note** ——— This clock must be the same clock as the PCLK on the External APBCOM for Integrated Cortex-M DAP module, as domain crossing is not handled inside the Debug Splitter. ———
CLK_INT	Used by the bridge modules on their respective sides.
CLK_EXT	

2.3 Hardware components

In a typical configuration, the SDC-600 Secure Debug Channel contains one External component and one Internal APBCOM component, with one or more COM asynchronous bridges in between.

The SDC-600 contains the following:

- An Internal APBCOM, which is used by the servicing agent and recognized as a peripheral device. The internal side contains the following interfaces:
  - An IRQ output, which can send an interrupt to the servicing agent.
  - An APB4 slave.
  - Two Q-Channel interfaces: one for clock and one for power.
  - Two COM wire interfaces: one for the Rx direction and one for the Tx direction.
- An External component, which is used on the debugger side and recognized as a special access port (COM-AP) or as a CoreSight peripheral device (APBCOM) by the debugger. This component has three different variants depending on the debug infrastructure version:
  - The External APBCOM component, which is used in ADIv6-compliant systems.
  - The External APBCOM for Integrated Cortex-M DAP component, which is used in Integrated Cortex-M DAP systems. This component is a CoreSight ROM table that includes COM functionality.
  - The External COM-AP component, which is used in ADIv5.2-compliant systems.

  The external side contains the following interfaces:
  - Powerup and Reboot Request interfaces, which can request power for the internal side and the optional bridges as well as to send a request to reboot the servicing agent.
  - A **DP_ABORT** input in ADIv6-compliant systems. An abort request causes the external side to terminate an ongoing transaction and free up its APB slave interface. In ADIv5.2-compliant systems, DAPBUS is used instead.

    ——— **Note** ———

    The Integrated Cortex-M DAP variant does not have this interface.

    ———————————

  - An APB4 slave. In ADIv5.2-compliant systems, DAPBUS is used instead.
  - A Q-Channel interface for the clock.
  - Two COM wire interfaces: one for the Rx direction and one for the Tx direction.
  - An Authentication interface, which is only available in the APBCOM for Integrated Cortex-M DAP component.
- COM asynchronous bridges, which transfer data between two different clock or power domains. The COM asynchronous bridges have the following interfaces:
  - Two COM wire interfaces on each side: one for the Rx direction and one for the Tx direction.
  - Two clock LPIs.
  - The Indirect Bridge also has two power LPIs.
- The Debug Splitter, which is required in a debug subsystem that cannot be extended by an additional Access Port (AP) or in a system that uses Integrated Cortex-M DAP. The Debug Splitter intercepts the AP downstream interface, and splits it into a downstream interface for the SDC-600 and another downstream interface for the rest of the system. The Debug Splitter contains:
  - A target slave multiplexer interface, which provides one slave interface and three master interfaces.
  - A low-latency synchronous interface bridge, which includes a slave target interface and an APB4 master interface.
  - A default slave interface.

  ——— **Note** ———

  The Debug Splitter component is not needed if the SDC-600 is implemented in an extendable debug system that is ADIv6 or ADIv5.2 compliant.

  ———————————

The following table shows which components are required for each variant:

Component	ADIv6	ADIv5.2	Integrated Cortex-M DAP
External APBCOM `sdc600_apbcom_ext`	Required	-	-
COM-AP `sdc600_comap`	-	Required	-
External APBCOM for Integrated Cortex-M DAP `sdc600_apbcom_ext_rom`	-	-	Required
Internal APBCOM `sdc600_apbcom_int`	Required	Required	Required

COM asynchronous bridge
`sdc600_comasyncbridge_direct_half_ext`
`sdc600_comasyncbridge_direct_half_int`
`sdc600_comasyncbridge_indirect_half_ext`
`sdc600_comasyncbridge_indirect_half_int`
	Optional	Optional	Optional

Debug Splitter `sdc600_debugsplitter`	-	-	Required

This section contains the following subsections:

2.3.1 External APBCOM

This is the external-side APBCOM component for ADIv6-compliant systems, such as Arm CoreSight SoC-600.

The APBCOM component provides APB access to a duplex *COM Wire Interface* (CWI) link for low-bandwidth communication. This component is used on the debugger side and recognized as a CoreSight peripheral device by the debugger. It has Powerup and Reboot Request interfaces to request power for the internal side or to send a request to reboot the servicing agent.

This component uses an APB bus. The following figure shows the port connections:

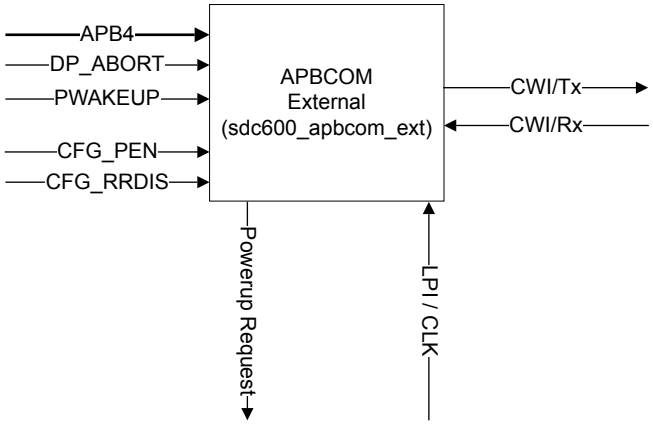

Figure 2-4 External APBCOM for ADIv6-compliant systems

See *3.2 Control and Status Register Map for sdc600_apbcom_ext* on page 3-36 for information about the address map configuration.

2.3.2 COM-AP

This is the external-side COM-AP component for ADIv5.2-compliant systems, such as Arm CoreSight SoC-400.

This component provides access to a duplex COM Wire Interface (CWI) link for low-bandwidth communication. The COM-AP is used on the debugger side and recognized as a special access port by the debugger. It has Powerup and Reboot Request interfaces to request power for the internal side or to send a request to reboot the servicing agent.

This component uses a DAP bus. The following figure shows the port connections:

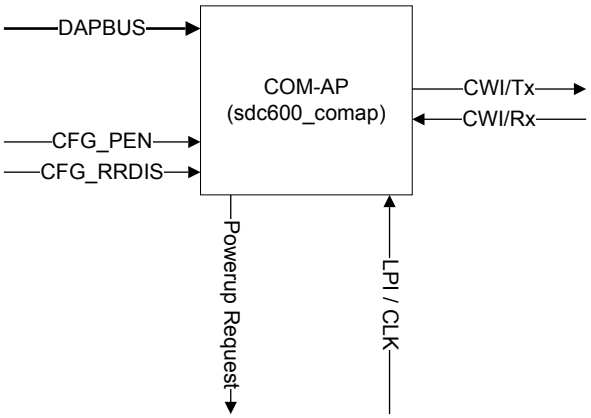

Figure 2-5 COM-AP for ADIv5.2-compliant systems

See *3.3 Control and Status Register Map for sdc600_comap* on page 3-37 for information about the address map configuration.

2.3.3 External APBCOM for systems with Integrated Cortex-M DAP

This is the external-side APBCOM component for systems with Integrated Cortex-M DAP.

This component provides APB access to a duplex COM Wire Interface (CWI) link for low-bandwidth communication.

It is a CoreSight ROM Table that includes COM functionality and is recognized by the debugger as such. The ROM table entry is used in the CoreSight discovery process, and would normally point to the debug components of the attached processor core. The single entry in this ROM table must be configured to point to the target processor core's ROM table.

It has Powerup and Reboot Request interfaces to request power for the internal side or to send a request to reboot the servicing agent.

This component uses an APB bus. The following figure shows the port connections:

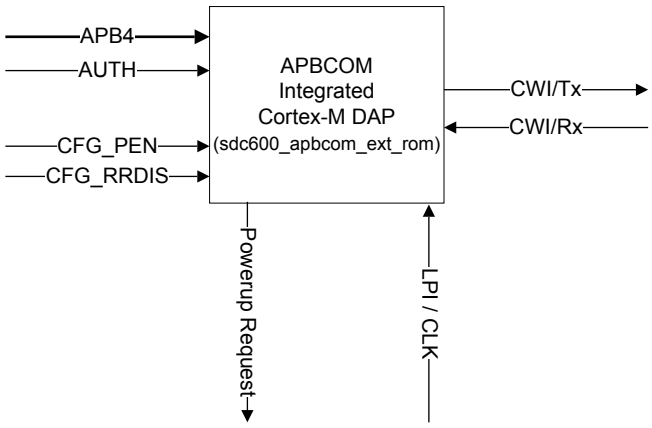

Figure 2-6 External APBCOM for systems with Integrated Cortex-M DAP

See *3.4 Control and Status Register Map for sdc600_apbcom_ext_rom* on page 3-39 for information about the address map configuration.

2.3.4 Internal APBCOM

The Internal APBCOM is the internal-side APBCOM component.

This component provides APB access to a duplex *COM Wire Interface* (CWI) link for low-bandwidth communication. It is used on the internal side and recognized as a CoreSight peripheral device by the servicing agent. It has a Power LPI interface to manage power and an IRQ interface to inform the servicing agent that an event occured.

This component uses an APB bus. The following figure shows the port connections:

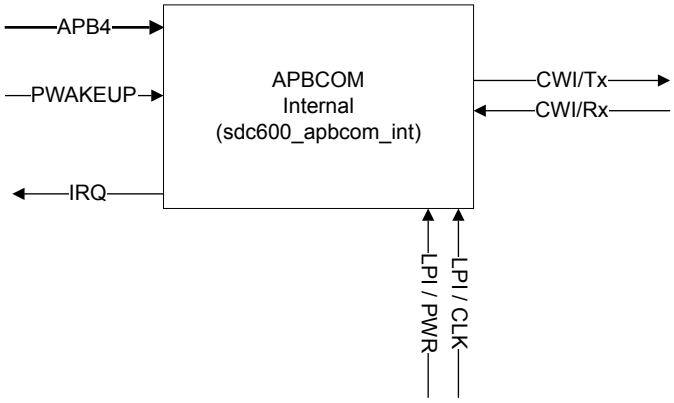

Figure 2-7 Internal APBCOM

The Internal APBCOM uses the same address map configuration as the External APBCOM. See *3.2 Control and Status Register Map for sdc600_apbcom_ext* on page 3-36 for information.

2.3.5 COM asynchronous bridge

The SDC-600 COM asynchronous bridge module is used as a connection between the External component and the Internal APBCOM when they are in different power or clock domains.

The bridge synchronizes the inputs coming from the clock domain of the other side. This module transfers and synchronizes data bytes between different domains using four-way handshake signals.

The bridge is available in two versions:

Direct COM Asynchronous bridge

For this area-optimized bridge, the data is not registered on either side of the bridge. The sideband signals are not registered on the sending side, but only synchronized on the receiving side of the bridge.

It has two CWIs on each side of the domains: one for the Rx and one for the Tx.

This bridge has two clock LPI interfaces, one for each clock domain.

Note

If there are multiple domain crossings, you must use Indirect Bridges.

The following figure shows the high-level blocks and transactions between the components when using a Direct Bridge:

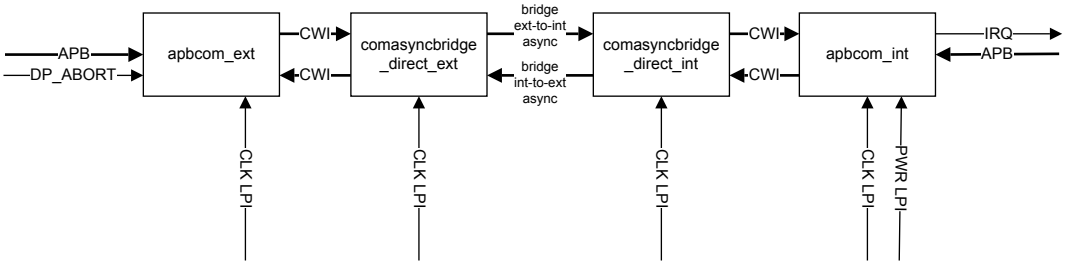

Figure 2-8 COM Asynchronous Direct Bridge high-level block diagram

Indirect COM Asynchronous bridge

For the Indirect Bridge, data is registered on both sides of the bridge. Sideband signals are all registered on the sending side and synchronized on the receiving side of the bridge. This is done to simplify constraining the CWI and floorplanning at the domain crossings.

The Indirect Bridge can be instantiated multiple times on a CWI channel if passing multiple domains. This bridge has two clock LPI interfaces and two power LPI interfaces to facilitate integration with multiple domain crossings.

The following figure shows the high-level blocks and transactions between the components when using an Indirect Bridge:

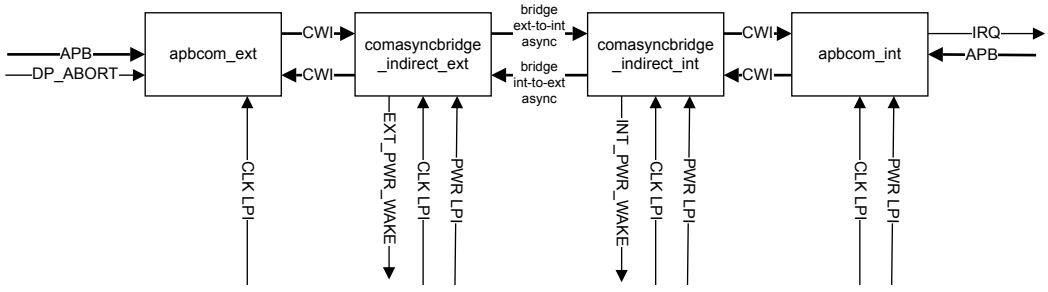

Figure 2-9 COM Asynchronous Indirect Bridge high-level block diagram

2.3.6 Debug Splitter

The Debug Splitter is a module that is placed between the AHB master port of the Integrated Cortex-M DAP and the SLV (debug) port of the processor core in non-AHB-Lite compliant mode.

Note

- The Debug Splitter is not used in extendable ADIv6- or ADIv5.2-compliant debug subsystems.
- The Debug Splitter can only be used with processor cores with Integrated Cortex-M DAP and can only be placed between the AHB master port of the Integrated Cortex-M DAP and the SLV (debug) port of the processor core.

The Debug Splitter module provides:

- Access control on the Cortex-M debug interface.
- Additional master interfaces to add another slave next to the processor core's debug port.
- A Cortex-M Debug slave interface for the Debug Access Port of the Integrated Cortex-M DAP on its debugger side.

On its target system side, it has:

- A Cortex-M Debug master interface.
- An APB4 master interface to the SDC-600.
- An authentication input interface, through which the target system can enable the intercepted Cortex-M Debug master interface.

The Debug Splitter contains the following components:

- A slave multiplexer.
- A default slave.
- A synchronous interface bridge.

The following figure shows the functional blocks of the Debug Splitter.

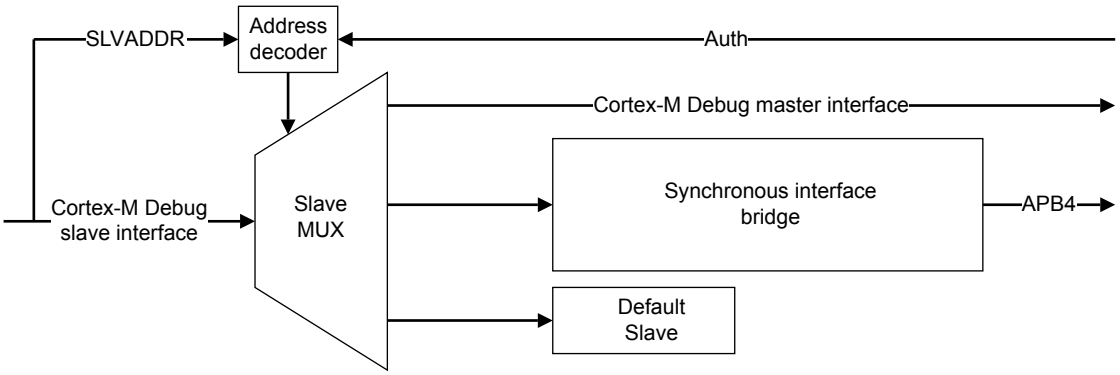

Figure 2-10 Debug Splitter block diagram

Slave multiplexer

The slave multiplexer has three ports:

- One for the path towards the processor core.
- One for SDC-600 through the synchronous interface bridge.
- One for the default slave.

Note

- If debug is not enabled, transactions are forwarded towards the default slave instead of the processor core, which terminates in an error. The path is therefore blocked by the address decoder when the debug is not enabled on the Authentication interface signals.
- The multiplexer forwards all non-32-bit accesses that are addressed to the SDC-600 to the default slave.

Default slave

The default slave has a slave interface where only a few signals are taking part in the error response generation method.

The default slave conforms to default slave behavior as described in *ARM® AMBA® 5 AHB Protocol Specification*.

Synchronous interface bridge

The synchronous interface bridge connects a low-bandwidth APB4 peripheral device to the Cortex-M Debug slave interface.

The synchronous interface bridge has the following interfaces:

- A slave initiator interface to connect to the Cortex-M Debug master interface.
- An APB4 master interface to connect to the sdc600_apbcom_ext_rom.

Note

The clocks are common.

2.4 SDC-600 Interfaces

This section provides an overview of the interfaces that are used in the SDC-600 Secure Debug Channel.

The SDC-600 uses the following bus standards:

- APB4 for all programming interfaces (except for COM-AP) and register access infrastructure.
- DAPBUS for the COM-AP programming interfaces and register access infrastructure.
- Cortex-M Debug for the AHB Access Port in the Integrated Cortex-M DAP and the processor debug port in non-AHB compliant mode.The Debug Splitter also has master and slave interfaces of this type.
- Q-Channel for Low-Power Interfaces.

This section contains the following subsections:

2.4.1 Clock and power control interfaces

The External component and the Internal APBCOM can be located in different clock and power domains.

APBCOM/COM-AP LPI

The External components (SDC-600 APBCOM, APBCOM for Integrated Cortex-M DAP, and COM-AP) have a clock LPI Q-Channel to communicate to the clock controller.

The Internal APBCOM has a clock LPI Q-Channel and also a power LPI Q-Channel to communicate with the power controller.

COM asynchronous bridge LPI

The SDC-600 COM asynchronous bridges have two clock LPI Q-Channels, which communicate with the clock controller.

The Indirect CWI asynchronous bridge module additionally has two power LPI Q-Channels, which communicate with the power controller.

Powerup and Reboot Request interfaces

When the External component is on and the Internal APBCOM is off, the External side can request power for the Internal APBCOM and detects when power is granted.

The powerup request is maintained for the duration of communication activity, such as authentication, and removed when finished.

The SDC-600 External component can send remote reset requests to the reset controller to request the servicing agent to reboot. If the Internal APBCOM is located in the same reset domain as the target agent, it and the affected bridge components are also reset.

2.4.2 Interface to the External components

This section describes the interface to the External components for each variant.

Interface to the External APBCOM

For ADIv6-compliant systems, the APB interface connects the External APBCOM to the *Debug Port* (DP) through APB infrastructure components. It is preferable for the External APBCOM to be high in the APB infrastructure hierarchy.

Note

- The SDC-600 can be connected lower in the hierarchy for Arm CoreSight SoC-600 implementations. However, in this case all elements on the route to the External APBCOM must be accessible without any authentication. Otherwise, the communication is blocked.
- See *3.2 Control and Status Register Map for sdc600_apbcom_ext* on page 3-36 for address map configuration.

The External APBCOM also has a **DP_ABORT** signal. An abort request causes the External APBCOM to terminate the current transaction. Abort requests only affect APB write transactions to the Data Blocking Register (DBR) stalled by a transaction still in progress on the CWI Tx interface. Other transactions, which are guaranteed to complete in a finite amount of time, ignore the abort and are allowed to complete normally.

When an abort occurs:

- The aborted transaction returns an APB error response and sets the TRINPROG bit in the Status Register.
- If SR.TRINPROG is set, any further writes to DR or DBR registers return error responses until the bit is cleared. The TRINPROG bit is cleared when the transaction that was in progress on the CWI Tx interface when the abort request was received is completed.
- Transactions on the CWI are not affected.

Interface to the COM-AP

For ADIv5.2-compliant systems, the DAPBUS interface connects the COM-AP to the Debug Port (DP) through the DAPBUS Interconnect (DAPIC).

The COM-AP has the **DAPABORT** signal, which is part of the DAPBUS interface and has the same functionality as the **DP_ABORT** with APB.

Note

Arm recommends that you connect the SDC-600 to the DAPIC in the hierarchy for Arm CoreSight SoC-400 implementations. All elements on the route to the COM-AP must be accessible without any authentication. Otherwise, the communication is blocked.

See *3.3 Control and Status Register Map for sdc600_comap* on page 3-37 for address map configuration.

Interface to the APBCOM for Integrated Cortex-M DAP

For Integrated Cortex-M DAP-compliant systems, the APBCOM for Integrated Cortex-M DAP is connected to the Integrated Cortex-M DAP module through the Debug Splitter. **DP_ABORT** is not implemented in Integrated Cortex-M DAP systems.

See *3.4 Control and Status Register Map for sdc600_apbcom_ext_rom* on page 3-39 for address map configuration.

2.4.3 COM Wire Interface

For full-duplex communication, two *COM Wire Interfaces* (CWIs) are implemented between the Internal APBCOM and the External modules of the SDC-600. This allows you to connect the Tx and Rx ports.

When the External component and Internal APBCOM are not in the same clock domain, synchronization is required. This is accomplished with a bridge component that converts the VALID/READY signaling style to the REQ/ACK four-way handshake style. In this case, one or more COM asynchronous bridge modules are placed between the External and Internal sides.

2.4.4 APB interface from the servicing agent to the Internal APBCOM

The interface from the servicing agent to the Internal APBCOM is APB.

The SDC-600 can generate an IRQ from the Internal APBCOM component, if enabled. Both Tx and Rx IRQ generation is implemented.

A Tx IRQ is generated if it is enabled and any of the following occurs:

- Tx FIFO is empty.
- Tx FIFO has less than a certain number of bytes remaining to be processed as specified by ICSR.TXFIL.

  ———— **Note** ————

  If ICSR.TXFIL is set higher than 1, which is the Tx FIFO depth, a Tx IRQ will be continuously generated.

  ————————————

An Rx IRQ is generated if it is enabled and any of the following occurs:

- RxEngine FIFO has at least the number of bytes ready as specified in ICSR.RXFIL.
- RxEngine FIFO is full. This occurs when the number of bytes stored reaches the value defined by FIDRXR.RXFD.

  ———— **Note** ————

  Rx IRQ generation is enabled by default. When the link establishment is initiated by the external side, an IRQ is generated. Setting ICSR.RXFIL higher than the Rx FIFO depth (FIDRXR_RXFD) results in the same behavior as setting it to the same value as FIDRXR_RXFD.

  ————————————

Chapter 3
Programmer's model

This chapter provides a description of the address map and registers of the SDC-600.

It contains the following sections:

3.1 About this programmers model

This section provides general information about SDC-600 register properties.

The following information applies to Arm 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. Unused address locations are RES0. 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.
- Access type is described as follows:

  RW
  Read and write.

  RO
  Read only.

  WO
  Write only.

  RAZ
  Read as zero.

  WI
  Writes ignored.

The programmer’s models of the External and Internal APBCOMs are the same. Only the relevant capability registers have different content according to what features are required on each side.

Note

The ITSTATUS and ITCTRL registers are used for the abort functionality and are therefore not implemented on the Internal APBCOM.

SDC-600 implements one-deep FIFOs in both Tx and Rx directions, which simplifies its programmer’s model and behavior.

The control and status registers are slightly different for each External variant:
- ADIv6 variant : The control and status registers have an offset of `0xD00` according to ADIv6.
- ADIv5.2 variant : The control and status registers have an offset of `0x00` according to ADIv5.2.
- Integrated Cortex-M DAP variant : This variant is a Class `0x9` ROM table with COM Port functionality built in. ROM table entries start at `0x000` according to the ADI specification. The COM Port functionality-related registers have an offset of `0xD00`.

3.2 Control and Status Register Map for sdc600_apbcom_ext

This variant can be used with ADIv6 debug subsystems, such as Arm CoreSight SoC-600 implementations.

The following figure shows the register map.

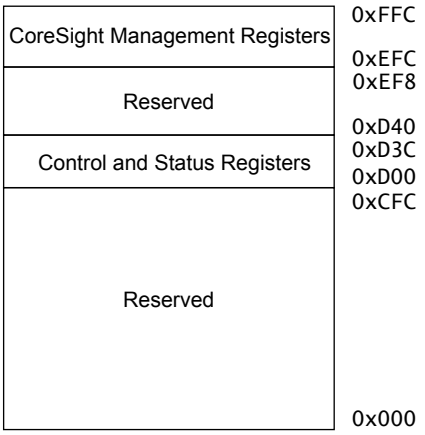

Figure 3-1 Control and Status Register map for ADIv6 debug subsystems

The following table shows the registers offset from the base memory address.

Table 3-1 Control and Status Register table for ADIv6 debug subsystems

Offset	Description
0xFFC-0xEFC	CoreSight Management Registers
0xEF8-0xD40	Reserved
0xD3C-0xD00	SDC-600 Control and Status Registers
0xCFC-0x000	Reserved

3.3 Control and Status Register Map for sdc600_comap

This variant can be used with ADIv5.2 debug subsystems, such as Arm CoreSight SoC-400 implementations.

The following figure shows the register map.

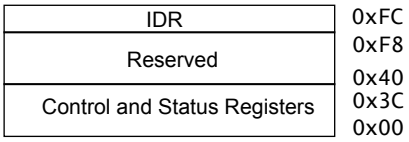

Figure 3-2 Control and Status Register map for ADIv5.2 debug subsystems

The following table shows the registers offset from the base memory address.

Table 3-2 Control and Status Register table for ADIv5.2 debug subsystems

Offset	Description
`0xFC`	*3.3.1 Identification Register* on page 3-37
`0xF8-0x40`	Reserved
`0x3C-0x00`	SDC-600 Control and Status Registers

This section contains the following subsection:

- *3.3.1 Identification Register* on page 3-37.

3.3.1 Identification Register

This register is used to identify the Access Port.

The IDR characteristics are:

Usage constraints
The value of this register is `0x0476_2000`.

Configurations
Available in the ADIv5.2 variant.

Attributes
32-bit read-only register.

The following figure shows the bit assignments.

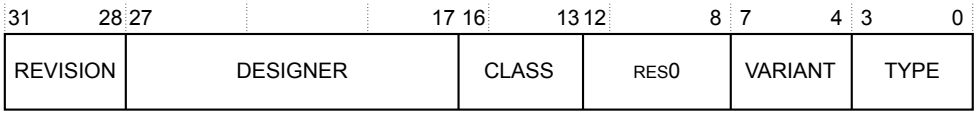

Figure 3-3 IDR bit assignments

The following table shows the bit assignments.

Table 3-3 IDR bit assignments

Bits	Name	Function	
[31:28]	REVISION	`0x0`	Revision 0.
[27:17]	DESIGNER	`0x23B`	Arm.

Table 3-3 IDR bit assignments (continued)

Bits	Name	Function	
[16:13]	CLASS	0b0001	COM Access Port.
[12:8]	-	RES0.	
[7:4]	VARIANT	0b0	COM-AP.
[3:0]	TYPE	0b0	COM-AP.

3.4 Control and Status Register Map for sdc600_apbcom_ext_rom

This variant can be used with systems with Integrated Cortex-M DAP.

The following figure shows the register map.

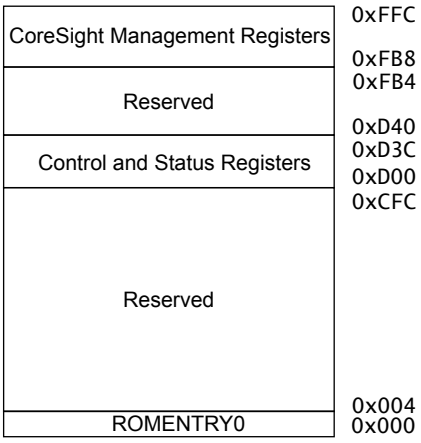

Figure 3-4 Control and Status Register map for systems with Integrated Cortex-M DAP

The following table shows the registers offset from the base memory address.

Table 3-4 Control and Status Register table for systems with Integrated Cortex-M DAP

Offset	Description
0xFFC-0xFB8	CoreSight Management Registers
0xFB4-0xD40	Reserved
0xD3C-0xD00	SDC-600 Control and Status Registers
0xCFC-0x004	Reserved
0x000	ROMENTRY0

This section contains the following subsection:

- *3.4.1 ROMENTRY0, Class 0x9 ROM Table entry* on page 3-39.

3.4.1 ROMENTRY0, Class 0x9 ROM Table entry

This entry is used to point to the ROM table of the target system.

The ROMENTRY0 characteristics are:

Usage constraints

This entry shows the base address of the ROM table of the target system, relative to the base address of this ROM table. It is defined by a configuration parameter. Bits [1:0] are updated according to the authentication status.

Configurations

Available in the Integrated Cortex-M DAP variant.

Attributes

32-bit read-only.

The following figure shows the bit assignments.

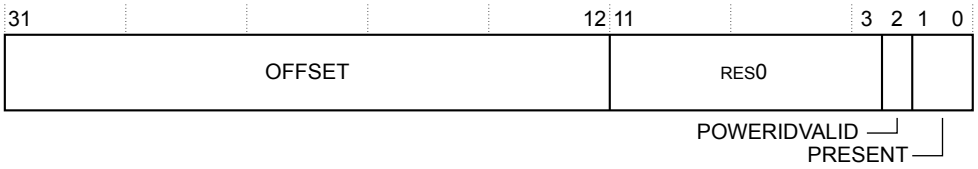

Figure 3-5 ROMENTRY0 bit assignments

The following table shows the bit assignments.

Table 3-5 ROMENTRY0 bit assignments

Bits	Name	Function
[31:12]	OFFSET	ROM_ENTRY0[31:12] Defined by the parameter `ROM_ENTRY0`. These are the upper bits of the relative address of the ROM table of the target system.
[11:3]	-	RES0.
[2]	POWERIDVALID	**0** No standard power control.
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. This field can have one of the following values: `0b10` The ROM entry is not present, and this ROMENTRY<n> is not the final entry in a ROM Table with fewer than the maximum number of entries. This value is set when all debug is disabled on the authentication interface. `0b11` The ROM Entry is present. This value is set when any debug is allowed on the authentication interface.

3.5 Control and Status Register Map for sdc600_apbcom_int

The control and status registers of the sdc600_apbcom_int are the same as for sdc600_apbcom_ext.

See *3.2 Control and Status Register Map for sdc600_apbcom_ext* on page 3-36 for more information.

3.6 Control and Status Registers

The following are the common control and status registers of the COM Port modules.

Note

Refer to the Register Map for each variant for the initial offset.

Table 3-6 Control and Status registers

Offset	Type	Size	Register	Description
0x00	RO	32	VIDR	*3.6.1 Version ID Register* on page 3-42
0x08	RO	32	FIDTXR	*3.6.2 Feature ID TxEngine Register* on page 3-43
0x0C	RO	32	FIDRXR	*3.6.3 Feature ID RxEngine Register* on page 3-44
0x10	RW	32	ICSR	*3.6.4 Interrupt Control Status Register* on page 3-45
0x20	RW	32	DR	*3.6.5 Data Register* on page 3-47
0x2C	RW	32	SR	*3.6.6 Status Register* on page 3-48
0x30	RW	32	DBR	*3.6.7 Data Blocking Register* on page 3-50
0x3C	RW	32	SR	*3.6.6 Status Register* on page 3-48

This section contains the following subsections:

- *3.6.1 Version ID Register* on page 3-42.
- *3.6.2 Feature ID TxEngine Register* on page 3-43.
- *3.6.3 Feature ID RxEngine Register* on page 3-44.
- *3.6.4 Interrupt Control Status Register* on page 3-45.
- *3.6.5 Data Register* on page 3-47.
- *3.6.6 Status Register* on page 3-48.
- *3.6.7 Data Blocking Register* on page 3-50.

3.6.1 Version ID Register

This register provides information about the architecture version of the COM Port.

The VIDR characteristics are:

Usage constraints

There are no usage constraints.

Configurations

Available in all SDC-600 configurations.

Attributes

32-bit read-only memory-mapped register located at offset 0x00.

This register resets to 0x00

The following figure shows the bit assignments.

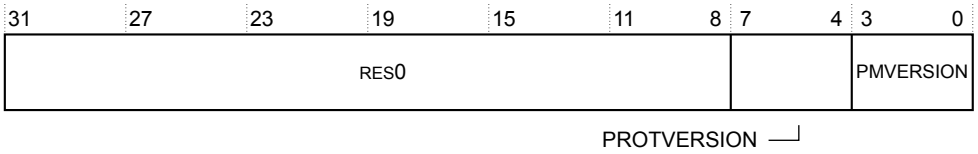

Figure 3-6 VIDR bit assignments

The following table shows the bit assignments.

Table 3-7 VIDR bit assignments

Bits	Name	Function
[31:8]	-	RES0.
[7:4]	PROTVERSION	APBCOM Protocol version. The value of this field is: `0x0` APBCOM Protocol version 0 implemented.
[3:0]	PMVERSION	APBCOM Programmers model version. The value of this field is: `0x0` APBCOM Programmers model version 0 implemented.

3.6.2 Feature ID TxEngine Register

This register provides information about the features implemented in the COM Port TxEngine.

The FIDTXR characteristics are:

Usage constraints
There are no usage constraints.

Configurations
Available in all SDC-600 configurations.

Attributes
32-bit read-only memory-mapped register located at offset `0x08`.

The following figure shows the bit assignments.

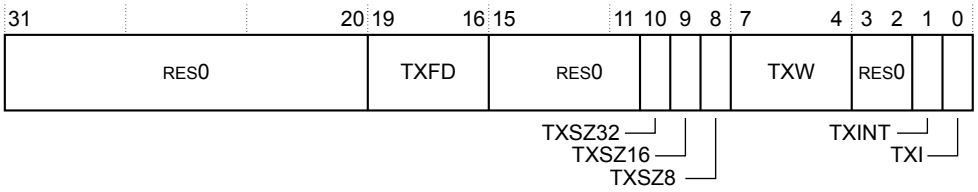

Figure 3-7 FIDTXR bit assignments

The following table shows the bit assignments.

Table 3-8 FIDTXR bit assignments

Bits	Name	Function
[31:20]	-	RES0.
[19:16]	TXFD	TxEngine FIFO depth. The defined value of this field is: `0x0` TxEngine FIFO has a capacity of 1 byte.
[15:11]	-	RES0.
[10]	TXSZ32	TxEngine 32-bit write support. The defined value of this bit is: `0x1` TxEngine supports 32-bit wide writes.
[9]	TXSZ16	TxEngine 16-bit write support. The defined value of this bit is: `0x0` TxEngine does not support 16-bit wide writes.

Table 3-8 FIDTXR bit assignments (continued)

Bits	Name	Function
[8]	TXSZ8	TxEngine 8-bit write support. The defined value of this bit is: `0x0` TxEngine does not support 8-bit wide writes.
[7:4]	TXW	TxEngine Width. Indicates the implemented width of the TxEngine. The defined value of this field is: `0x1` 1-byte wide TxEngine. The upper bytes must be the NULL flag (`0xAF`). For example, to write `0x12` to the TxEngine, the APB write data is `0xAFAFAF12`.
[3:2]	-	RES0.
[1]	TXINT	Indicates whether the TxEngine generates interrupts. The defined values of this bit are: `0x0` TxEngine interrupts not implemented. `0x1` TxEngine interrupts implemented. **Note** Only sdc600_apbcom_int has interrupt capabilities. The external COM Port modules do not have interrupt functionality. If the TxEngine interrupts are implemented, the following are implemented: • ICSR.TXFIL • ICSR.TXFIS
[0]	TXI	Indicates whether the TxEngine is implemented. The defined value of this bit is: `0x1` TxEngine implemented.

3.6.3 Feature ID RxEngine Register

This register provides information about the features implemented in the COM Port RxEngine.

The FIDRXR characteristics are:

Usage constraints
There are no usage constraints.

Configurations
Available in all SDC-600 configurations.

Attributes
32-bit read-only memory-mapped register located at offset `0x0C`.

The following figure shows the bit assignments.

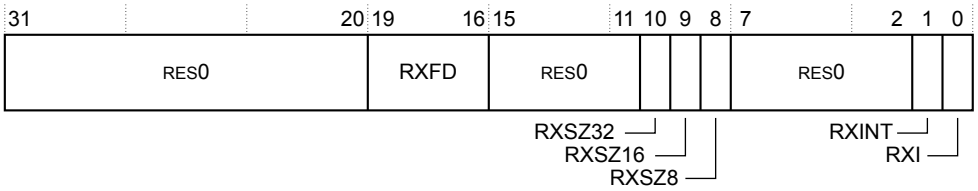

Figure 3-8 FIDRXR bit assignments

The following table shows the bit assignments.

Table 3-9 FIDRXR bit assignments

Bits	Name	Function
[31:20]	-	RES0.
[19:16]	RXFD	RxEngine FIFO depth. The defined value of this field is: `0x0` RxEngine FIFO has a capacity of 1 byte.
[15:11]	-	RES0.
[10]	RXSZ32	RxEngine 32-bit read support. The defined value of this bit is: `0x1` RxEngine supports 32-bit wide reads.
[9]	RXSZ16	RxEngine 16-bit read support. The defined value of this bit is: `0x0` RxEngine does not support 16-bit wide reads.
[8]	RXSZ8	RxEngine 8-bit read support. The defined value of this bit is: `0x0` RxEngine does not support 8-bit wide reads.
[7:2]	-	RES0.
[1]	RXINT	Indicates whether the RxEngine generates interrupts. The defined values of this bit are: `0x0` RxEngine interrupts not implemented. `0x1` RxEngine interrupts implemented. **Note** Only sdc600_apbcom_int has interrupt capabilities. The external COM Port modules do not have interrupt functionality. If the RxEngine interrupts are implemented, the following are implemented: • ICSR.RXFIL • ICSR.RXFIS
[0]	RXI	Indicates whether the RxEngine is implemented. The defined value of this bit is: `0x1` RxEngine implemented.

3.6.4 Interrupt Control Status Register

This register controls and indicates the current status of the COM Port interrupt functionality.

The ICSR characteristics are:

Usage constraints
There are no usage constraints.

Configurations
Only present in the Internal APBCOM. Otherwise this register is RES0.

Attributes
32-bit read/write memory-mapped register located at offset `0x10`.

The following figure shows the bit assignments.

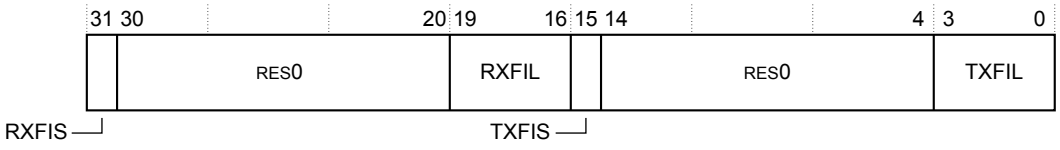

Figure 3-9 ICSR bit assignments

The following table shows the bit assignments.

Table 3-10 ICSR bit assignments

Bits	Name	Function
[31]	RXFIS	RxEngine FIFO interrupt status. The possible values of this bit are: `0x0` RxEngine FIFO interrupt has not occurred. `0x1` RxEngine FIFO interrupt has occurred. This bit is RES0 in the External COM Port modules. This bit is read/write-one-to-clear. This bit cannot be cleared while there is a triggering condition, such as RXFIL = 1 and RxFIFO is full. This bit resets to `0x0`.
[30:20]	-	RES0.
[19:16]	RXFIL	RxEngine FIFO interrupt level select. The possible values of this field are: `0x0` RxEngine FIFO interrupts disabled. `0x1-0xF` Generate RxEngine FIFO interrupt when the RxEngine FIFO is full. This occurs when the number of bytes stored reaches the value defined by FIDRXR.RXFD. This field is RES0 in the External COM Port modules. This field is read/write. This field resets to `0x1`.
[15]	TXFIS	TxEngine FIFO interrupt status. The possible values of this bit are: `0x0` TxEngine FIFO interrupt has not occurred. `0x1` TxEngine FIFO interrupt has occurred. This bit is RES0 in the External COM Port modules. This bit is read/write-one-to-clear. This bit cannot be cleared while there is a triggering condition, such as TXFIL = 1 and TxFIFO is empty. This bit resets to `0x0`.

Table 3-10 ICSR bit assignments (continued)

Bits	Name	Function
[14:4]	-	RES0.
[3:0]	TXFIL	TxEngine FIFO interrupt level select. The possible values of this field are: `0x0` TxEngine FIFO interrupts disabled. `0x1-0xF` Generate TxEngine FIFO interrupt when the TxEngine FIFO has less than the specified number of bytes remaining to process. **Note** If TXFIL is set to a value higher than 1, the IRQ will be continuously generated. See *2.4.4 APB interface from the servicing agent to the Internal APBCOM* on page 2-32. This field is RES0 in the external COM Port modules. This field is read/write. This field resets to `0x0`.

3.6.5 Data Register

This register is used to send data via the TxEngine and receive data from the RxEngine.

The DR characteristics are:

Usage constraints
Upper bytes must be the NULL flag (`0xAF`) on writes.

Configurations
Available in all SDC-600 configurations.

Attributes
32-bit read/write memory-mapped register located at offset `0x20`.

The DR and DBR operate identically, except on writes where more data is written to the TxEngine than the COM Port can accept. In this case:

- When writing to the DR, the write access completes with an OK response and an overflow error is logged in SR.TXOE.
- When writing to the DBR, the write access stalls until there is sufficient space.

This register is RW.

This register resets to `0xAFAFAFAF`.

The following figure shows the bit assignments.

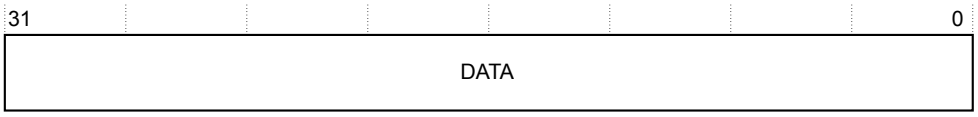

Figure 3-10 DR bit assignments

The following table shows the bit assignments.

Table 3-11 DR bit assignments

Bits	Name	Function
[31:0]	DATA	Data transfer. Only 32-bit accesses are supported to DR. On writes: • Transfers the lowest byte into the TxEngine FIFO for transmission. • The upper bytes must be written with the NULL Flag byte value. • The TxEngine ignores the NULL Flag byte value. • If TxEngine FIFO is not empty and the byte to be transmitted to the TxEngine is not a NULL Flag byte, a TxEngine Overflow error occurs and SR.TXOE is set to 1. The byte is discarded by the TxEngine. • If SR.TXOE or SR.TXLE is 1, then writes are ignored. • Write accesses complete immediately. On reads: • Returns one byte from the RxEngine FIFO at the lowest byte. • If the RxEngine FIFO is empty, the lowest byte returns the NULL Flag byte. The upper bytes always return the NULL Flag byte value of `0xAF`. • Read accesses complete immediately.

3.6.6 Status Register

This register indicates the current status of the COM Port component.

The SR characteristics are:

Usage constraints
There are no usage constraints.

Configurations
Available in all SDC-600 configurations.

Attributes
32-bit read/write memory-mapped register located at:
- Offset `0x2C`
- Offset `0x3C`

SR is aliased in more than one location. Accesses to any SR location access a single physical SR.

The following figure shows the bit assignments.

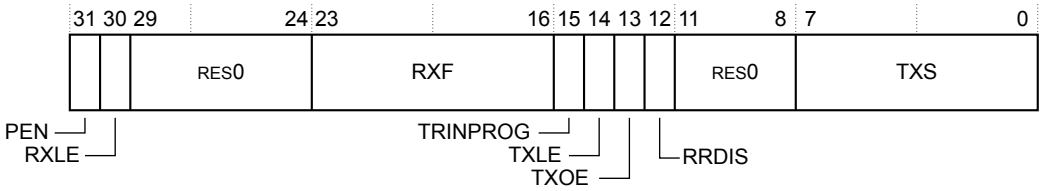

Figure 3-11 SR bit assignments

The following table shows the bit assignments.

Table 3-12 SR bit assignments

Bits	Name	Function
[31]	PEN	COM Port component enabled status. The defined values of this bit are: `0x0` Component is disabled. • Writes to DR and DBR are ignored. • Reads of DR and DBR behave as if the RxEngine FIFO is empty. `0x1` Component is enabled. This bit is read-only. Reading this bit returns the value of the **CFG_PEN** input on the External components and `0x1` on the Internal APBCOM.
[30]	RXLE	RxEngine link error detected. This bit is RES0 because no link error can occur in the SDC-600 RxEngine.
[29:24]	-	RES0.
[23:16]	RXF	RxEngine FIFO fill level. The possible values of this field are: `0x00` RxEngine has no data. `0x01` RxEngine has at least 1 byte available to read. This field is read-only. This field resets to `0x00`.
[15]	TRINPROG	Transfer in progress. This bit is set when a transaction is aborted via **DP_ABORT**. The possible values of this bit are: `0x0` No transaction in progress. `0x1` An input transaction has been aborted but the internal operation of that transaction, or a previous transaction, is still in progress. This bit is RES0 in the External APBCOM for Integrated Cortex-M DAP and the Internal APBCOM components. This bit resets to `0x0`.
[14]	TXLE	TxEngine link error detected. The possible values of this bit are: `0x0` No link error detected. `0x1` A link error has been detected in the TxEngine. This bit is set to 1 if: • The link is lost during a data transfer because the remote Rx interface module is not operating. A LERR flag byte is inserted into the local Rx FIFO. • One or more bytes written to the Tx FIFO are discarded because the link to the remote Rx interface module is not operating. A LERR flag byte is inserted into the local Rx FIFO. This bit is read/write-one-to-clear. This bit resets to `0x0`.

Table 3-12 SR bit assignments (continued)

Bits	Name	Function
[13]	TXOE	TxEngine FIFO overflow. The possible values of this bit are: `0x0` No overflow logged. `0x1` At least one byte written to TxEngine could not be accepted and has been lost. This bit is read/write-one-to-clear. This bit resets to `0x0`.
[12]	RRDIS	Remote reboot requests disabled. The defined values of this bit are: `0x0` Remote Reboot requests enabled. `0x1` Remote Reboot requests disabled. When this bit is 1, the TxEngine discards any LPH2RR Flag bytes written to the TxEngine. This bit is read-only. Reading this bit returns the value of the CFG_RRDIS input after synchronization in the External components and `0x1` in the Internal APBCOM.
[11:8]	-	RES0.
[7:0]	TXS	TxEngine FIFO space. The defined values of this field are: `0x00` TxEngine has no space for new data. `0b1` TxEngine has at least 1 byte available. This field resets to `0x01`. This field is read-only.

3.6.7 Data Blocking Register

This register is used to send data via the TxEngine and receive data from the RxEngine.

The DBR characteristics are:

Usage constraints
Upper bytes must be the NULL flag (`0xAF`) on writes.

Configurations
Available in all SDC-600 configurations.

Attributes
32-bit read/write memory-mapped register located at offset `0x30`.

The DR and DBR operate identically, except on writes where more data is written to the TxEngine than the COM Port can accept. In this case:

- When writing to the DR, the write access completes with an OK response and an overflow error is logged in SR.TXOE.
- When writing to the DBR, the write access stalls until there is sufficient space.

The following figure shows the bit assignments.

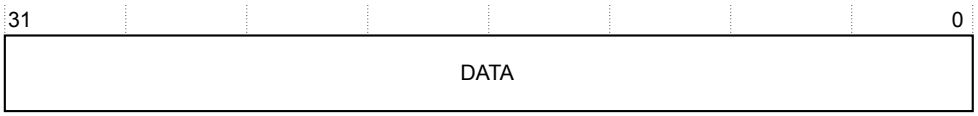

Figure 3-12 DBR bit assignments

The following table shows the bit assignments.

Table 3-13 DBR bit assignments

Bits	Name	Function
[31:0]	DATA	Data transfer. Only 32-bit access size is supported to DBR. On writes: • Transfers the lowest byte into the TxEngine FIFO for transmission. • The upper bytes must be written with the NULL Flag byte value. • The TxEngine ignores the NULL Flag byte value. • If TxEngine FIFO is not empty and the byte to be transmitted to the TxEngine is not a NULL Flag byte, the write to the DBR stalls until the FIFO empties. • If SR.TXOE or SR.TXLE is 1, then writes are ignored. • If SR.TXLE is 1, then writes are ignored and any outstanding write access is terminated with an OK response. On reads: • Returns one byte from the RxEngine FIFO at the lowest byte. • If the RxEngine FIFO is empty, the lowest byte returns the NULL Flag byte. The upper bytes always return the NULL Flag byte. • Read accesses complete immediately.

3.7 CoreSight™ Management Registers

Here are the CoreSight Management registers.

These registers are the same for each variant with the following exceptions:

- The COM-AP component does not implement CoreSight Management Registers.
- DEVID is only present in the Integrated Cortex-M DAP variant.
- ITCTRL and ITSTATUS are RAZ/WI in the Internal and Integrated Cortex-M DAP variants.
- CLAIMSET and CLAIMCLR are RAZ/WI in the Integrated Cortex-M DAP variant.
- AUTHSTATUS is RAZ/WI in variants other than Integrated Cortex-M DAP variant.

Note

For more information about these registers, see *ARM® CoreSight™ Architecture Specification v3.0*.

Table 3-14 CoreSight Management registers

Offset	Type	Size	Register	Reset	Description
0xEFC	RO	32	ITSTATUS	0	*3.7.1 Integration Mode Status Register* on page 3-53
0xF00	RW	32	ITCTRL	0	*3.7.2 Integration Mode Control Register* on page 3-54
0xFA0	RW RO[a]	32	CLAIMSET	0x3 0[a]	Claim Tag Set Register
0xFA4	RW RO[a]	32	CLAIMCLR	0	Claim Tag Clear Register
0xFA8	RO	32	DEVAFF0	0	Device Affinity Registers
0xFAC	RO	32	DEVAFF1	0	
0xFB0	RO	32	LAR	0	Software Lock Access Register
0xFB4	RO	32	LSR	0	Software Lock Status Register
0xFB8	RO	32	AUTHSTATUS	0 0xAA[a]	Authentication Status Register
0xFBC	RO	32	DEVARCH	0x4770_0A57	Device Architecture Register: CoreSight SDC-600
				0x4770_0AF7	Device Architecture Register: CoreSight ROM Table
0xFC0	RO	32	DEVID2	0	Device Configuration Register 2
0xFC4	RO	32	DEVID1	0	Device Configuration Register 1
0xFC8	RO	32	DEVID	0x00 0x40[a]	*3.7.3 Device ID Register* on page 3-54
0xFCC	RO	32	DEVTYPE	0	Device Type Identifier Register
0xFD0	RO	32	PIDR4	0x04[b]	Peripheral ID Register
0xFD4	RO	32	PIDR5	0	Peripheral ID Register
0xFD8	RO	32	PIDR6	0	Peripheral ID Register
0xFDC	RO	32	PIDR7	0	Peripheral ID Register

Table 3-14 CoreSight Management registers (continued)

Offset	Type	Size	Register	Reset	Description
0xFE0	RO	32	PIDR0	0xB9[b]	Peripheral ID Register
0xFE4	RO	32	PIDR1	0xEF[b]	Peripheral ID Register
0xFE8	RO	32	PIDR2	0x0B[b]	Peripheral ID Register
0xFEC	RO	32	PIDR3	0x00[b]	Peripheral ID Register
0xFF0	RO	32	CIDR0	0x0D	Component ID Register
0xFF4	RO	32	CIDR1	0x90	Component ID Register
0xFF8	RO	32	CIDR2	0x05	Component ID Register
0xFFC	RO	32	CIDR3	0xB1	Component ID Register

This section contains the following subsections:

- *3.7.1 Integration Mode Status Register* on page 3-53.
- *3.7.2 Integration Mode Control Register* on page 3-54.
- *3.7.3 Device ID Register* on page 3-54.

3.7.1 Integration Mode Status Register

This register is used for the Integration Test DP Abort Status.

The ITSTATUS characteristics are:

Usage constraints
There are no usage constraints.

Configurations
Only available in the ADIv6 variant.

Attributes
32-bit read-only memory-mapped register.

The following figure shows the bit assignments.

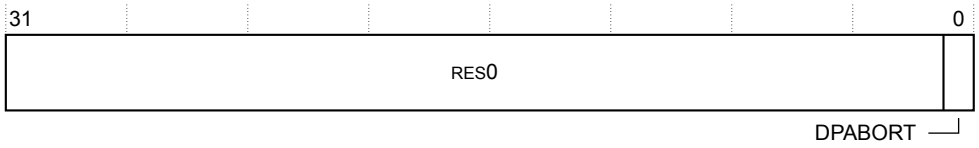

Figure 3-13 ITSTATUS bit assignments

The following table shows the bit assignments.

[a] This value is only for sdc600_apbcom_ext_rom.
[b] The reset value for sdc600_apbcom_ext_rom is parameter dependant.

Table 3-15 ITSTATUS bit assignments

Bits	Name	Function
[31:1]	-	RES0.
[0]	DPABORT	In integration testing mode, when ITCTRL.IME = 1, this bit latches to 1 on the rising edge of **DP_ABORT**. The possible values of this bit are: `0x0` No rising edge of **DP_ABORT** has been detected. `0x1` Rising edge has been detected on **DP_ABORT**. This bit is cleared on APB read from this register. If **DP_ABORT** rises in the same cycle when an APB read of the ITSTATUS register is received, the APB read takes priority and the register is cleared as a result. In mission mode, when ITCTRL.IME = 0, this register is RAZ/WI. This bit resets to `0x0`.

3.7.2 Integration Mode Control Register

This register is used to enable the Integration Mode.

The ITCTRL characteristics are:

Usage constraints
There are no usage constraints.

Configurations
Only available in the ADIv6 variant.

Attributes
32-bit read/write memory-mapped register.

The following figure shows the bit assignments.

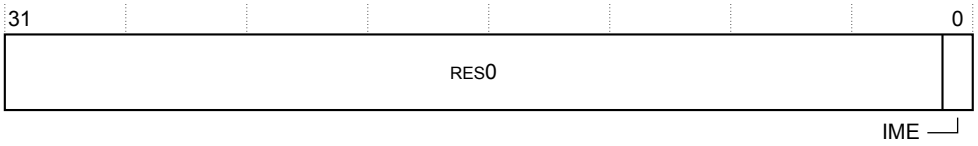

Figure 3-14 ITCTRL bit assignments

The following table shows the bit assignments.

Table 3-16 ITCTRL bit assignments

Bits	Name	Function
[31:1]	-	RES0.
[0]	IME	Integration Mode Enable. When set, the APBCOM enters integration mode, which allows integration testing to be performed. The possible values of this bit are: `0x0` Functional mode. `0x1` Integration test mode. This bit resets to `0x0`.

3.7.3 Device ID Register

This register is used to indicate if the COM Port function is included in the ROM table.

The DEVID characteristics are:

Usage constraints
There are no usage constraints.

Configurations
Only available for sdc600_apbcom_ext_rom.

Attributes
32-bit read/write memory-mapped register.

The following figure shows the bit assignments.

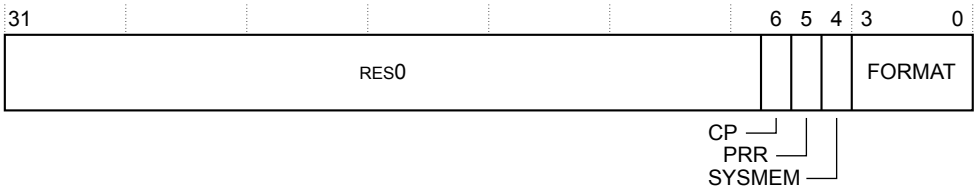

Figure 3-15 DEVID bit assignments

The following table shows the bit assignments.

Table 3-17 DEVID bit assignments

Bits	Name	Function
[31:7]	-	RES0.
[6]	CP	COM Port functionality present. The offset range `0xD00-0xD7C` contains COM Port functionality. Offset `0xD00` indicates the COM Port programmers model. The value of this bit is: `0b1` COM Port functionality present.
[5]	PRR	Powerup Request functionality included. The value of this bit is: `0b0` Powerup Request functionality not included.
[4]	SYSMEM	System memory present. Indicates whether system memory is present on the bus that connects to the ROM table. The value of this bit is: `0b1` System memory is present on the bus that connects to the ROM table.
[3:0]	FORMAT	ROM format. The value of this field is: `0x0` 32-bit format 0.

Appendix A
Signal descriptions

This chapter describes the signals of the SDC-600.

It contains the following sections:

A.1 Signals for the sdc600_apbcom_ext

The following tables show the signals of the ADIv6 External APBCOM component.

Table A-1 External APBCOM, system signals

Name	Direction	Description
PCLK	Input	The main clock. All signal timings are related to the rising edge.
PRESET_N	Input	Active-LOW reset signal.
CFG_RRDIS	Input	Remote reboot request disable signal from the application that drives the read-only SR.RRDIS bit of the external COM Port components. It provides control to an application or the lifecycle state management of the chip to prevent rebooting the target system through the SDC-600 when it is no longer needed. CDC synchronizer flops are implemented inside the module for this input, so changing the value of this signal during operation is allowed. If this feature is not required, Arm recommends that you disable it so that this feature is not exposed to the debugger.
CFG_PEN	Input	COM Port enable that drives the read-only SR.PEN bit of the External COM Port components. If not set: • Writes to DR and DBR are ignored. • Reads of DR and DBR behave as if the RxEngine FIFO would be empty. It is expected to be a static configuration, therefore no CDC synchronizer flops are implemented inside the module for this input.

Table A-2 External APBCOM, APB4 signals

Name	Direction	Description
PADDR_S[11:0]	Input	Address bus.
PWRITE_S	Input	Indicates the direction of the data transfer. When HIGH, write is enabled. When LOW, read is enabled.
PSEL_S	Input	When HIGH, the APB slave is selected.
PENABLE_S	Input	Starts the APB transfer one cycle after **PSEL**.
PWDATA_S [31:0]	Input	The data input of the APB slave when **PWRITE_S** is HIGH.
PRDATA_S [31:0]	Output	The selected slave drives this bus when **PWRITE_S** is LOW.
PREADY_S	Output	The APBCOM uses this ready signal to extend the transfer. Its inactive state is LOW.
PSLVERR_S	Output	The APBCOM uses this error signal to indicate that an error occurred during the transfer and the transfer was aborted.

Table A-3 External APBCOM, CWI-RX signals

Name	Direction	Description
RX_VALID	Input	Data valid.
RX_READY	Output	Data ready. Indicates the completion of a transfer.
RX_DATA[7:0]	Input	Data line from Internal APBCOM's TxEngine.

Table A-3 External APBCOM, CWI-RX signals (continued)

Name	Direction	Description
RX_LINKEST	Input	Link established signal from Internal APBCOM.
RX_LINKUP	Output	Linkup indicated to Internal APBCOM.

Table A-4 External APBCOM, CWI-TX signals

Name	Direction	Description
TX_VALID	Output	Data valid.
TX_READY	Input	Data ready. Indicates the completion of a transfer.
TX_DATA[7:0]	Output	Data line to Internal APBCOM's RxEngine.
TX_LINKEST	Output	Link established signal to Internal APBCOM.
TX_LINKUP	Input	Linkup indicated from Internal APBCOM.

Table A-5 External APBCOM, Powerup Request signals

Name	Direction	Description
REMPUR	Output	Remote powerup request to the power control unit.
REMPUA	Input	Remote powerup acknowledge from the power control unit.
REMRR	Output	Remote reboot request to the power control unit.
REMRA	Input	Remote reboot acknowledge from the power control unit.

Table A-6 External APBCOM, Q-Channel for clock signals

Name	Direction	Description
CLK_QREQ_N	Input	Asynchronous quiescence request signal for the clock.
CLK_QACCEPT_N	Output	When LOW, the quiescent request is accepted for the clock.
CLK_QDENY	Output	When HIGH, the quiescent request is denied for the clock.
CLK_QACTIVE	Output	When HIGH, indicates to the controller that the component requires the clock.

Table A-7 External APBCOM, miscellaneous signals

Name	Direction	Description
DP_ABORT	Input	When an APB transfer takes more time than expected, the DP is able to abort the transfer. When HIGH, the transfer either finishes normally in deterministic short time or is aborted.
PWAKEUP_S	Input	Wakeup request. Indicates that a new transaction is placed or about to be placed on the APB slave interface. This signal is assumed to stay asserted for the entire length of **PSEL_S**. It is guaranteed by the APB master to be glitch-free, and is used to drive **CLK_QACTIVE**.

A.2 Signals for the sdc600_comap

The following tables show the signals of the ADIv5.2 COM-AP component.

Table A-8 COM-AP, system signals

Name	Direction	Description
DAPCLK	Input	The main clock. All signal timings are related to the rising edge.
DAPRESET_N	Input	Active-LOW reset signal.
CFG_RRDIS	Input	Remote reboot request disable signal from the application that drives the read-only SR.RRDIS bit of the external COM Port components. It provides control to an application or the lifecycle state management of the chip to prevent rebooting the target system through the SDC-600 when it is no longer needed. CDC synchronizer flops are implemented inside the module for this input, so changing the value of this signal during operation is allowed. If this feature is not required, Arm recommends that you disable it so that this feature is not exposed to the debugger.
CFG_PEN	Input	COM Port enable that drives the read-only SR.PEN bit of the External COM Port components. If not set: • Writes to DR and DBR are ignored. • Reads of DR and DBR behave as if the RxEngine FIFO would be empty. It is expected to be a static configuration, therefore no CDC synchronizer flops are implemented inside the module for this input.

Table A-9 COM-AP, DAPBUS signals

Name	Direction	Description
DAPCADDR_S [7:0]	Input	Compressed address bus from the DP.
DAPWRITE_S	Input	Indicates the direction of the data transfer. When HIGH, write is enabled. When LOW, read is enabled.
DAPSEL_S	Input	When HIGH, the DAP Bus slave is selected.
DAPENABLE_S	Input	Starts the DAP Bus transfer when **DAPWRITE_S** is HIGH.
DAPWDATA_S [31:0]	Input	The data input of the DAP Bus slave when **DAPWRITE_S** is HIGH.
DAPRDATA_S [31:0]	Output	The selected slave drives this bus when **DAPWRITE_S** is LOW.
DAPREADY_S	Output	The COM-AP uses this ready signal to extend the transfer. Its inactive state is LOW.
DAPSLVERR_S	Output	The COM-AP uses this error signal to indicate that an error occurred during the transfer and the transfer was aborted.
DAPABORT_S	Input	When a transfer takes more time than expected, the DAP is able to abort the transfer. When HIGH, the transfer either finishes normally in deterministic short time or is aborted.

Table A-10 COM-AP, CWI-RX signals

Name	Direction	Description
RX_VALID	Input	Data valid.
RX_READY	Output	Data ready. Indicates the completion of a transfer.
RX_DATA[7:0]	Input	Data line from the Internal APBCOM's TxEngine.
RX_LINKEST	Input	Link established signal from Internal APBCOM.
RX_LINKUP	Output	Linkup indicated to Internal APBCOM.

Table A-11 COM-AP, CWI-TX signals

Name	Direction	Description
TX_VALID	Output	Data valid.
TX_READY	Input	Data ready. Indicates the completion of a transfer.
TX_DATA[7:0]	Output	Data line to Internal APBCOM's RxEngine.
TX_LINKEST	Output	Link established signal to Internal APBCOM.
TX_LINKUP	Input	Linkup indicated from Internal APBCOM.

Table A-12 COM-AP, Powerup Request signals

Name	Direction	Description
REMPUR	Output	Remote powerup request to the power control unit.
REMPUA	Input	Remote powerup acknowledge from the power control unit.
REMRR	Output	Remote reboot request to the power control unit.
REMRA	Input	Remote reboot acknowledge from the power control unit.

Table A-13 COM-AP, Q-Channel for clock signals

Name	Direction	Description
CLK_QREQ_N	Input	Asynchronous quiescence request signal for the clock.
CLK_QACCEPT_N	Output	When LOW, the quiescent request is accepted for the clock.
CLK_QDENY	Output	When HIGH, the quiescent request is denied for the clock.
CLK_QACTIVE	Output	When HIGH, indicates to the controller that the component requires the clock.

A.3 Signals for the sdc600_apbcom_ext_rom

The following tables show the signals of the Integrated Cortex-M DAP External APBCOM component.

Table A-14 External APBCOM Integrated Cortex-M DAP, system signals

Name	Direction	Description
PCLK	Input	The main clock. All signal timings are related to the rising edge.
PRESET_N	Input	Active-LOW reset signal.
CFG_RRDIS	Input	Remote reboot request disable signal from the application that drives the read-only SR.RRDIS bit of the external COM Port components. It provides control to an application or the lifecycle state management of the chip to prevent rebooting the target system through the SDC-600 when it is no longer needed. CDC synchronizer flops are implemented inside the module for this input, so changing the value of this signal during operation is allowed. If this feature is not required, Arm recommends that you disable it so that this feature is not exposed to the debugger.
CFG_PEN	Input	COM Port enable that drives the read-only SR.PEN bit of the External COM Port components. If not set: • Writes to DR and DBR are ignored. • Reads of DR and DBR behave as if the RxEngine FIFO would be empty. It is expected to be a static configuration, therefore no CDC synchronizer flops are implemented inside the module for this input.

Table A-15 External APBCOM Integrated Cortex-M DAP, APB4 signals

Name	Direction	Description
PADDR_S[11:0]	Input	Address bus.
PWRITE_S	Input	Indicates the direction of the data transfer. When HIGH, write is enabled. When LOW, read is enabled.
PSEL_S	Input	When HIGH, the APB slave is selected.
PENABLE_S	Input	Starts the APB transfer one cycle after **PSEL**.
PWDATA_S [31:0]	Input	The data input of the APB slave when **PWRITE_S** is HIGH.
PRDATA_S [31:0]	Output	The selected slave drives this bus when **PWRITE_S** is LOW.
PREADY_S	Output	The APBCOM uses this ready signal to extend the transfer. Its inactive state is LOW.

Table A-16 External APBCOM Integrated Cortex-M DAP, CWI-RX signals

Name	Direction	Description
RX_VALID	Input	Data valid.
RX_READY	Output	Data ready. Indicates the completion of a transfer.
RX_DATA[7:0]	Input	Data line from Internal APBCOM's TxEngine.

Table A-16 External APBCOM Integrated Cortex-M DAP, CWI-RX signals (continued)

Name	Direction	Description
RX_LINKEST	Input	Link established signal from Internal APBCOM.
RX_LINKUP	Output	Linkup indicated to Internal APBCOM.

Table A-17 External APBCOM Integrated Cortex-M DAP, CWI-TX signals

Name	Direction	Description
TX_VALID	Output	Data valid.
TX_READY	Input	Data ready. Indicates the completion of a transfer.
TX_DATA[7:0]	Output	Data line to Internal APBCOM's RxEngine.
TX_LINKEST	Output	Link established signal to Internal APBCOM.
TX_LINKUP	Input	Linkup indicated from Internal APBCOM.

Table A-18 External APBCOM Integrated Cortex-M DAP, Powerup Request signals

Name	Direction	Description
REMPUR	Output	Remote powerup request to the power control unit.
REMPUA	Input	Remote powerup acknowledge from the power control unit.
REMRR	Output	Remote reboot request to the power control unit.
REMRA	Input	Remote reboot acknowledge from the power control unit.

Table A-19 External APBCOM Integrated Cortex-M DAP, authentication interface signals

Name	Direction	Description
SPNIDEN	Input	Secure non-invasive debug enable.
SPIDEN	Input	Secure invasive debug enable.
NIDEN	Input	Non-secure non-invasive debug enable.
DBGEN	Input	Non-secure invasive debug enable.

See the *Authentication Interface* section of the *ARM® CoreSight™ Architecture Specification v3.0* for more information about the authentication interface signals.

Table A-20 External APBCOM Integrated Cortex-M DAP, Q-Channel for clock signals

Name	Direction	Description
CLK_QREQ_N	Input	Asynchronous quiescence request signal for the clock.
CLK_QACCEPT_N	Output	When LOW, the quiescent request is accepted for the clock.
CLK_QDENY	Output	When HIGH, the quiescent request is denied for the clock.
CLK_QACTIVE	Output	When HIGH, indicates to the controller that the component requires the clock.

A.4 Signals for the sdc600_apbcom_int

This section describes the signals used with the internal APBCOM.

Table A-21 Internal APBCOM system signals

Name	Direction	Description
PCLK	Input	The main clock. All signal timings are related to the rising edge.
PRESET_N	Input	Active-LOW reset signal.

Table A-22 Internal APBCOM APB4 signals

Name	Direction	Description
PADDR_S[11:0]	Input	Address bus.
PWRITE_S	Input	Indicates the direction of the data transfer. When HIGH, write is enabled. When LOW, read is enabled.
PSEL_S	Input	When HIGH, the APB slave is selected.
PENABLE_S	Input	Starts the APB transfer one cycle after **PSEL**.
PWDATA_S [31:0]	Input	The data input of the APB slave when **PWRITE_S** is HIGH.
PRDATA_S [31:0]	Output	The selected slave drives this bus when **PWRITE_S** is LOW.
PREADY_S	Output	The Internal APBCOM uses this ready signal to extend the transfer if it is in low-power state during the transfer initiation. Its inactive state is LOW.

Table A-23 Internal APBCOM CWI-RX signals

Name	Direction	Description
RX_VALID	Input	Data valid.
RX_READY	Output	Data ready. Indicates the completion of a transfer.
RX_DATA[7:0]	Input	Data line from the External component's TxEngine.
RX_LINKEST	Input	Link established signal from the External component.
RX_LINKUP	Output	Linkup indicated to the External component. The reset value is LOW.

Table A-24 Internal APBCOM CWI-TX signals

Name	Direction	Description
TX_VALID	Output	Data valid.
TX_READY	Input	Data ready. Indicates the completion of a transfer.
TX_DATA[7:0]	Output	Data line to the External component's RxEngine. The reset value is LOW.
TX_LINKEST	Output	Link established signal to the External component. The reset value is LOW.
TX_LINKUP	Input	Linkup indicated from the External component.

Table A-25 Internal APBCOM Q-Channel for clock and power signals

Name	Direction	Description
PWR_QREQ_N	Input	Asynchronous quiescence request signal for power.
PWR_QACCEPT_N	Output	When LOW, the quiescent request is accepted for the power.
PWR_QDENY	Output	When HIGH, the quiescent request is denied for the power.
PWR_QACTIVE	Output	When HIGH, indicates to the Q-Channel interface that the component requires power.
CLK_QREQ_N	Input	Asynchronous quiescence request signal for the clock.
CLK_QACCEPT_N	Output	When LOW, the quiescent request is accepted for the clock.
CLK_QDENY	Output	When HIGH, the quiescent request is denied for the clock.
CLK_QACTIVE	Output	When HIGH, indicates to the Q-Channel interface that the component requires the clock.

Table A-26 Internal APBCOM miscellaneous signals

Name	Direction	Description
IRQ	Output	Interrupt signal to the servicing agent. The reset value is LOW.
PWAKEUP_S	Input	Wake up request. Indicates that a new transaction is placed or about to be placed on the APB slave interface. This signal is assumed to stay asserted for the entire length of **PSEL_S**. It is guaranteed by the APB master to be glitch-free, and is used to drive **CLK_QACTIVE**.

A.5 COM asynchronous bridge signals

This section describes the signals used with the bridge components.

This section contains the following subsections:

A.5.1 External indirect half bridge signals

The following tables show the signals for the external indirect half bridge.

Table A-27 External indirect half bridge system signals

Name	Direction	Description
CLK_EXT	Input	Clock for the external side of the bridge. All signal timings are related to the rising edge.
RESETN_EXT	Input	Active-LOW reset for CLK_EXT domain.

Table A-28 External indirect half bridge COM-RX signals

Name	Direction	Description
EXT_RX_DATA[7:0]	Input	Receive data from External component.
EXT_RX_VALID	Input	Data valid.
EXT_RX_READY	Output	Data accepted, ready delivered by bridge.
EXT_RX_LINKUP	Output	Linkup indicated by the Internal APBCOM to the External component.
EXT_RX_LINKEST	Input	Link established from the External component to the Internal APBCOM.

Table A-29 External indirect half bridge COM-TX signals

Name	Direction	Description
EXT_TX_DATA[7:0]	Output	Data transmitted to the External component.
EXT_TX_VALID	Output	Data valid.
EXT_TX_READY	Input	Data accepted.
EXT_TX_LINKUP	Input	Linkup indicated by the External component to the Internal APBCOM.
EXT_TX_LINKEST	Output	Link established from the Internal APBCOM to the External component.

Table A-30 External indirect half bridge asynchronous COM External to Internal signals

Name	Direction	Description
EXT_ASYNC_EI_REQ	Output	Transmit request handshake signal. The reset value is LOW.
EXT_ASYNC_EI_ACK	Input	Transmit acknowledge handshake signal.
EXT_ASYNC_EI_DATA[7:0]	Output	Transmit data. The reset value is LOW.
EXT_ASYNC_EI_LINKEST	Output	Link established from the External component to the Internal APBCOM. The reset value is LOW.
EXT_ASYNC_EI_LINKUP	Input	Linkup indicated by the Internal APBCOM to the External component.

Table A-31 External indirect half bridge asynchronous COM Internal to External signals

Name	Direction	Description
EXT_ASYNC_IE_REQ	Input	Receive data available handshake signal.
EXT_ASYNC_IE_ACK	Output	Receive data captured handshake signal. The reset value is LOW.
EXT_ASYNC_IE_DATA[7:0]	Input	Receive data.
EXT_ASYNC_IE_LINKEST	Input	Link established from the Internal APBCOM to the External component.
EXT_ASYNC_IE_LINKUP	Output	Linkup indicated by the External component to the Internal APBCOM. The reset value is LOW.

Table A-32 External indirect half bridge Clock Q-Channel signals

Name	Direction	Description
EXT_CLK_QREQ_N	Input	Asynchronous quiescence request signal for CLK_EXT.
EXT_CLK_QACCEPT_N	Output	When LOW, the quiescent request is accepted for CLK_EXT.
EXT_CLK_QDENY	Output	When HIGH, the quiescent request is denied for CLK_EXT.
EXT_CLK_QACTIVE	Output	When HIGH, indicates to the controller that the component requires the clock on CLK_EXT.

Table A-33 External indirect half bridge Power Q-Channel on Internal side signals

Name	Direction	Description
EXT_PWR_QREQ_N	Input	Asynchronous quiescence request signal for power down.
EXT_PWR_QACCEPT_N	Output	When LOW, the quiescent request is accepted for power down.
EXT_PWR_QDENY	Output	When HIGH, the quiescent request is denied for power down.
EXT_PWR_QACTIVE	Output	When HIGH, indicates to the Q-Channel interface that the component requires power.
EXT_PWR_WAKE	Output	Single signal power Q-Channel interface to wake the other side of the bridge on CWI activity.

A.5.2 Internal indirect half bridge signals

The following tables show the signals for the internal indirect half bridge.

Table A-34 Internal indirect half bridge system signals

Name	Direction	Description
CLK_INT	Input	Clock for the internal side of the bridge. All signal timings are related to the rising edge.
RESETN_INT	Input	Active-LOW reset for CLK_INT domain.

Table A-35 Internal indirect half bridge COM-TX signals

Name	Direction	Description
INT_TX_DATA[7:0]	Output	Transmit data to the Internal APBCOM.
INT_TX_VALID	Output	Data valid.
INT_TX_READY	Input	Data accepted.

Table A-35 Internal indirect half bridge COM-TX signals (continued)

Name	Direction	Description
INT_TX_LINKUP	Input	Linkup indicated by the Internal APBCOM to the External component.
INT_TX_LINKEST	Output	Link established from the External component to the Internal APBCOM.

Table A-36 Internal indirect half bridge COM-RX signals

Name	Direction	Description
INT_RX_DATA[7:0]	Input	Receive data from Internal APBCOM.
INT_RX_VALID	Input	Data valid.
INT_RX_READY	Output	Data accepted, ready delivered by bridge.
INT_RX_LINKUP	Output	Linkup indicated by the External component to the Internal APBCOM.
INT_RX_LINKEST	Input	Link established from the Internal APBCOM to the External component.

Table A-37 Internal indirect half bridge asynchronous COM External to Internal signals

Name	Direction	Description
INT_ASYNC_EI_REQ	Input	Receive data available handshake signal.
INT_ASYNC_EI_ACK	Output	Receive data captured handshake signal. The reset value is LOW.
INT_ASYNC_EI_DATA[7:0]	Input	Receive data.
INT_ASYNC_EI_LINKEST	Input	Link established from the External component to the Internal APBCOM.
INT_ASYNC_EI_LINKUP	Output	Linkup indicated by the Internal APBCOM to the External component. The reset value is LOW.

Table A-38 Internal indirect half bridge asynchronous COM Internal to External signals

Name	Direction	Description
INT_ASYNC_IE_REQ	Output	Transmit request handshake signal. The reset value is LOW.
INT_ASYNC_IE_ACK	Input	Transmit acknowledge handshake signal.
INT_ASYNC_IE_DATA[7:0]	Output	Transmitted data. The reset value is LOW.
INT_ASYNC_IE_LINKEST	Output	Link established from the Internal APBCOM to the External component. The reset value is LOW.
INT_ASYNC_IE_LINKUP	Input	Linkup indicated by the External component to the Internal APBCOM.

Table A-39 Internal indirect half bridge Clock Q-Channel signals

Name	Direction	Description
INT_CLK_QREQ_N	Input	Asynchronous quiescence request signal for CLK_INT.
INT_CLK_QACCEPT_N	Output	When LOW, the quiescent request is accepted for CLK_INT.
INT_CLK_QDENY	Output	When HIGH, the quiescent request is denied for CLK_INT.
INT_CLK_QACTIVE	Output	When HIGH, indicates to the controller that the component requires the clock on CLK_INT.

Table A-40 Internal indirect half bridge Power Q-Channel signals

Name	Direction	Description
INT_PWR_QREQ_N	Input	Asynchronous quiescence request signal for power down.
INT_PWR_QACCEPT_N	Output	When LOW, the quiescent request is accepted for power down.
INT_PWR_QDENY	Output	When HIGH, the quiescent request is denied for power down.
INT_PWR_QACTIVE	Output	When HIGH, indicates to the controller that the component requires power.
INT_PWR_WAKE	Output	Auxiliary signal that can be used by the system to wake up the domain. It is directly connected to **INT_PWR_QACTIVE**.

A.5.3 External direct half bridge signals

The following tables show the signals for the external direct half bridge.

Table A-41 External direct half bridge system signals

Name	Direction	Description
CLK_EXT	Input	Clock for the external side of the bridge. All signal timings are related to the rising edge.
RESETN_EXT	Input	Active-LOW reset for CLK_EXT domain.

Table A-42 External direct half bridge COM-RX signals

Name	Direction	Description
EXT_RX_DATA[7:0]	Input	Receive data from External component.
EXT_RX_VALID	Input	Data valid.
EXT_RX_READY	Output	Data accepted, ready delivered by bridge.
EXT_RX_LINKUP	Output	Linkup indicated by the Internal APBCOM to the External component.
EXT_RX_LINKEST	Input	Link established from the External component to the Internal APBCOM.

Table A-43 External direct half bridge COM-TX signals

Name	Direction	Description
EXT_TX_DATA[7:0]	Output	Data transmitted to the External component.
EXT_TX_VALID	Output	Data valid.
EXT_TX_READY	Input	Data accepted.
EXT_TX_LINKUP	Input	Linkup indicated by the External component to the Internal APBCOM.
EXT_TX_LINKEST	Output	Link established from the Internal APBCOM to the External component.

Table A-44 External direct half bridge asynchronous COM External to Internal signals

Name	Direction	Description
EXT_ASYNC_EI_REQ	Output	Transmit request handshake signal.
EXT_ASYNC_EI_ACK	Input	Transmit acknowledge handshake signal.
EXT_ASYNC_EI_DATA[7:0]	Output	Transmit data.

Table A-44 External direct half bridge asynchronous COM External to Internal signals (continued)

Name	Direction	Description
EXT_ASYNC_EI_LINKEST	Output	Link established from the External component to the Internal APBCOM.
EXT_ASYNC_EI_LINKUP	Input	Linkup indicated by the Internal APBCOM to the External component.

Table A-45 External direct half bridge asynchronous COM Internal to External signals

Name	Direction	Description
EXT_ASYNC_IE_REQ	Input	Receive data available handshake signal.
EXT_ASYNC_IE_ACK	Output	Receive data captured handshake signal.
EXT_ASYNC_IE_DATA[7:0]	Input	Receive data.
EXT_ASYNC_IE_LINKEST	Input	Link established from the Internal APBCOM to the External component.
EXT_ASYNC_IE_LINKUP	Output	Linkup indicated by the External component to the Internal APBCOM.

Table A-46 External direct half bridge Clock Q-Channel signals

Name	Direction	Description
EXT_CLK_QREQ_N	Input	Asynchronous quiescence request signal for CLK_EXT.
EXT_CLK_QACCEPT_N	Output	When LOW, the quiescent request is accepted for CLK_EXT.
EXT_CLK_QDENY	Output	When HIGH, the quiescent request is denied for CLK_EXT.
EXT_CLK_QACTIVE	Output	When HIGH, indicates to the controller that the component requires the clock on CLK_EXT.

A.5.4 Internal direct half bridge signals

The following tables show the signals for the internal direct half bridge.

Table A-47 Internal direct half bridge system signals

Name	Direction	Description
CLK_INT	Input	Clock for the internal side of the bridge. All signal timings are related to the rising edge.
RESETN_INT	Input	Active-LOW reset for CLK_INT domain.

Table A-48 Internal direct half bridge COM-TX signals

Name	Direction	Description
INT_TX_DATA[7:0]	Output	Transmit data to the Internal APBCOM.
INT_TX_VALID	Output	Data valid.
INT_TX_READY	Input	Data accepted.
INT_TX_LINKUP	Input	Linkup indicated by the Internal APBCOM to the External component.
INT_TX_LINKEST	Output	Link established from the External component to the Internal APBCOM.

Table A-49 Internal direct half bridge COM-RX signals

Name	Direction	Description
INT_RX_DATA[7:0]	Input	Receive data from Internal APBCOM.
INT_RX_VALID	Input	Data valid.
INT_RX_READY	Output	Data accepted, ready delivered by bridge.
INT_RX_LINKUP	Output	Linkup indicated by the External component to the Internal APBCOM.
INT_RX_LINKEST	Input	Link established from the Internal APBCOM to the External component.

Table A-50 Internal direct half bridge asynchronous COM External to Internal signals

Name	Direction	Description
INT_ASYNC_EI_REQ	Input	Receive data available handshake signal.
INT_ASYNC_EI_ACK	Output	Receive data capture handshake signal.
INT_ASYNC_EI_DATA[7:0]	Input	Receive data.
INT_ASYNC_EI_LINKEST	Input	Link established from the External component to the Internal APBCOM.
INT_ASYNC_EI_LINKUP	Output	Linkup indicated by the Internal APBCOM to the External component.

Table A-51 Internal direct half bridge asynchronous COM Internal to External signals

Name	Direction	Description
INT_ASYNC_IE_REQ	Output	Transmit request handshake signal.
INT_ASYNC_IE_ACK	Input	Transmit acknowledge handshake signal.
INT_ASYNC_IE_DATA[7:0]	Output	Transmitted data.
INT_ASYNC_IE_LINKEST	Output	Link established from the Internal APBCOM to the External component.
INT_ASYNC_IE_LINKUP	Input	Linkup indicated by the External component to the Internal APBCOM.

Table A-52 Internal direct half bridge Clock Q-Channel signals

Name	Direction	Description
INT_CLK_QREQ_N	Input	Asynchronous quiescence request signal for CLK_INT.
INT_CLK_QACCEPT_N	Output	When LOW, the quiescent request is accepted for CLK_INT.
INT_CLK_QDENY	Output	When HIGH, the quiescent request is denied for CLK_INT.
INT_CLK_QACTIVE	Output	When HIGH, indicates to the controller that the component requires the clock on CLK_INT.

A.6 Debug Splitter signals

The following tables show the signals of the Debug Splitter component.

Table A-53 Debug Splitter system signals

Name	Direction	Description
CLK	Input	The bus clock times all bus transfers. All signal timings are related to the rising edge.
RESET_N	Input	Active-LOW reset signal that resets the system and the bus.

Note

The SLV prefix in the following signals is specific to the Cortex-M Debug protocol. The _S and _M endings of the signal name indicate whether the signal belongs to the Slave or Master interface.

Table A-54 Debug Splitter Cortex-M Debug slave interface signals

Name	Direction	Description
SLVADDR_S[31:0]	Input	The 32-bit wide address bus.
SLVPROT_S[6:0]	Input	The protection control signals provide additional information about a bus access and are primarily intended for use by any module that wants to implement some level of protection. The signals indicate if the transfer is an opcode fetch or data access, and if the transfer is a privileged-mode access or user-mode access. For masters with a Memory Management Unit, these signals also indicate whether the current access is cacheable or bufferable.
SLVNONSEC_S	Input	Indicates if the current transfer is a Non-secure transfer or a Secure transfer.
SLVTRANS_S[1:0]	Input	Indicates the transfer type of the current transfer. `IDLE` or `NONSEQUENTIAL`.
SLVWDATA_S[31:0]	Input	The write data bus transfers data from the master to the slaves during write operations. The data bus width is fixed at 32 bits.
SLVWRITE_S	Input	Indicates the transfer direction. When HIGH, this signal indicates a write transfer and when LOW a read transfer. It has the same timing as the address signals.
SLVREADY_S	Output	When HIGH, the **HREADY** signal indicates to the master that the previous transfer is complete.
SLVRDATA_S[31:0]	Output	During read operations, the read data bus transfers data from the selected slave to the multiplexer. The multiplexer then transfers the data to the master. The data bus width is fixed at 32 bits.
SLVRESP_S	Output	The transfer response provides the master with additional information on the status of a transfer. LOW means OKAY and HIGH means ERROR.

Table A-55 Debug Splitter Cortex-M Debug master interface signals

Name	Direction	Description
SLVADDR_M[31:0]	Output	The 32-bit wide address bus.
SLVPROT_M[6:0]	Output	The protection control signals provide additional information about a bus access and are primarily intended for use by any module that wants to implement some level of protection. The signals indicate if the transfer is an opcode fetch or data access, and if the transfer is a privileged-mode access or user-mode access. For masters with a Memory Management Unit, these signals also indicate whether the current access is cacheable or bufferable.

Table A-55 Debug Splitter Cortex-M Debug master interface signals (continued)

Name	Direction	Description
SLVSIZE_M[1:0]	Output	Indicates the size of the transfer, which is byte, halfword, or word.
SLVNONSEC_M	Output	Indicates if the current transfer is a Non-secure transfer or a Secure transfer.
SLVTRANS_M[1:0]	Output	Indicates the transfer type of the current transfer. `IDLE` or `NONSEQUENTIAL`.
SLVWDATA_M[31:0]	Output	The write data bus transfers data from the master to the slaves during write operations. The data bus width is fixed at 32 bits.
SLVWRITE_M	Output	Indicates the transfer direction. When HIGH, this signal indicates a write transfer and when LOW a read transfer. It has the same timing as the address signals.
SLVREADY_M	Input	When HIGH, the **HREADY** signal indicates to the master that the previous transfer is complete.
SLVRDATA_M[31:0]	Input	During read operations, the read data bus transfers data from the selected slave to the multiplexer. The multiplexer then transfers the data to the master. The data bus width is fixed at 32 bits.
SLVRESP_M	Input	The transfer response provides the master with additional information on the status of a transfer. LOW means OKAY and HIGH means ERROR.

Table A-56 Debug Splitter APB4 master interface signals

Name	Direction	Description
PADDR_M[11:0]	Output	The APB address bus, which is driven by the peripheral bus bridge unit.
PSEL_M	Output	Indicates that the APB slave device is selected and that a data transfer is required.
PENABLE_M	Output	Indicates the second and subsequent cycles of an APB transfer.
PWRITE_M	Output	Indicates an APB Write-Access when HIGH and an APB read access when LOW.
PWDATA_M[31:0]	Output	This bus is driven by the peripheral bus bridge unit during write cycles when **PWRITE_M** is HIGH.
PRDATA_M[31:0]	Input	The selected APB slave drives this bus during read cycles when **PWRITE_M** is LOW.
PREADY_M	Input	The APB slave uses this signal to extend an APB transfer.
PSLVERR_M	Input	This signal indicates a transfer failure. **Note** Tie this input LOW when connecting to an APB slave device that does not provide a **PSLVERR** output, such as apbcom_ext_rom.

Table A-57 Authentication interface signals

Name	Direction	Description
SPNIDEN	Input	Secure non-invasive debug enable.
SPIDEN	Input	Secure invasive debug enable.
NIDEN	Input	Non-secure non-invasive debug enable.
DBGEN	Input	Non-secure invasive debug enable.

See the Authentication Interface section of the ARM CoreSight Architecture Specification v3.0 for more information about the authentication interface signals.

Appendix B
Revisions

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

It contains the following section:

- *B.1 Revisions* on page Appx-B-74.

B.1 Revisions

Lists the technical differences between released versions of the document.

Table B-1 Issue 0001-00

Change	Location	Affects
First release	-	-

Table B-2 Differences between issue 0001-00 and issue 0002-00

Change	Location	Affects
sdc600_debugsplitter signals.	*2.2 Clocks* on page 2-23 *A.6 Debug Splitter signals* on page Appx-A-71	r0p2
Bridge internal interface signals renamed.	*A.5.2 Internal indirect half bridge signals* on page Appx-A-66 *A.5.4 Internal direct half bridge signals* on page Appx-A-69	r0p2
An additional Power LPI Q-Channel interface on the Indirect Bridge.	*COM asynchronous bridge LPI* on page 2-31 *A.5.1 External indirect half bridge signals* on page Appx-A-65 *A.5.2 Internal indirect half bridge signals* on page Appx-A-66	r0p2
Power LPI interface removed from sdc600_comap, sdc600_apbcom_ext, and the sdc600_apbcom_ext_rom.	*APBCOM/COM-AP LPI* on page 2-31 *2.3.1 External APBCOM* on page 2-25 *2.3.2 COM-AP* on page 2-26 *2.3.3 External APBCOM for systems with Integrated Cortex-M DAP* on page 2-26	r0p2
EXT_PWR_WAKE signal added to both sides of Indirect Bridge.	*A.5.1 External indirect half bridge signals* on page Appx-A-65 *A.5.2 Internal indirect half bridge signals* on page Appx-A-66	r0p2
General restructuring of the document. The External components and variants have been separated both in the functional and signal descriptions.	Throughout document.	All versions.

Table B-3 Differences between issue 0002-00 and issue 0002-01

Change	Location	Affects
Signals ending in **n** renamed to **_N**	Throughout document.	r0p2

Table B-4 Differences between issue 0002-01 and issue 0002-02

Change	Location	Affects
Added defintions of terminology used throughout the document.	*1.1 About* on page 1-12 *1.3 Configurations* on page 1-14	All versions.
Clarified bridge definitions.	*1.4 Components* on page 1-15	All versions.
Reset Request Interface changed to Reboot Request Interface.	Throughout document.	All versions.
Updated the design process.	*1.6 Design Process* on page 1-17	All versions.

Table B-4 Differences between issue 0002-01 and issue 0002-02 (continued)

Change	Location	Affects
Updated the description of the Internal APBCOM.	*2.3.4 Internal APBCOM* on page 2-27	All versions.
Added Control and Status Register Map for the Internal APBCOM.	*3.5 Control and Status Register Map for sdc600_apbcom_int* on page 3-41	All versions.
Updated description for OFFSET field in ROMENTRY0 register.	*3.4.1 ROMENTRY0, Class 0x9 ROM Table entry* on page 3-39	All versions.
SYSMEM bit in DEVID register now set to 1.	*3.7.3 Device ID Register* on page 3-54	r0p2