Arm® CoreSight™ System-on-Chip SoC-600M

Revision: r1p1

Technical Reference Manual

Non-Confidential

Issue 00
101883_0101_00_en

Arm® CoreSight™ System-on-Chip SoC-600M
Technical Reference Manual

Release Information

Document history

Issue	Date	Confidentiality	Change
0000-00	13 December 2019	Non-Confidential	First release for r0p0 EAC
0000-01	9 January 2020	Non-Confidential	Second release for r0p0 EAC
0100-00	7 August 2020	Non-Confidential	First release for r1p0 REL
0101-00	15 July 2022	Non-Confidential	First release for r1p1 REL

Proprietary Notice

Confidentiality Status

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

Unrestricted Access is an Arm internal classification.

Product Status

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

Feedback

Arm welcomes feedback on this product and its documentation. To provide feedback on the product, create a ticket on https://support.developer.arm.com.

To provide feedback on the document, fill the following survey: https://developer.arm.com/documentation-feedback-survey.

Inclusive language commitment

Arm values inclusive communities. Arm recognizes that we and our industry have used language that can be offensive. Arm strives to lead the industry and create change.

This document includes language that can be offensive. We will replace this language in a future issue of this document.

To report offensive language in this document, email terms@arm.com.

Contents

1. Introduction

1.1 Product revision status

The `r`*`x`*`p`*`y`* identifier indicates the revision status of the product described in this manual, for example, `r`*`1`*`p`*`2`*, where:

r*x*	Identifies the major revision of the product, for example, r1.
p*y*	Identifies the minor revision or modification status of the product, for example, p2.

1.2 Intended audience

This book is written for the following audiences:

- Hardware and software engineers who want to incorporate CoreSight™ SoC-600M into their design and produce real-time instruction and data trace information from a SoC.
- Software engineers writing tools to use CoreSight™ SoC-600M.

This book assumes that readers are familiar with AMBA® bus design and JTAG methodology.

1.3 Conventions

The following subsections describe conventions used in Arm documents.

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: developer.arm.com/glossary.

Typographic conventions

Convention	Use
italic	Citations.
bold	Terms in descriptive lists, where appropriate.
`monospace`	Text that you can enter at the keyboard, such as commands, file and program names, and source code.

Convention	Use
`monospace underline`	A permitted abbreviation for a command or option. You can enter the underlined text instead of the full command or option name.
`<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	Terms that have specific technical meanings as defined in the *Arm® Glossary*. For example, IMPLEMENTATION DEFINED, IMPLEMENTATION SPECIFIC, UNKNOWN, and UNPREDICTABLE.

Caution

Recommendations. Not following these recommendations might lead to system failure or damage.

Warning

Requirements for the system. Not following these requirements might result in system failure or damage.

Danger

Requirements for the system. Not following these requirements will result in system failure or damage.

Note

An important piece of information that needs your attention.

Tip

A useful tip that might make it easier, better or faster to perform a task.

Remember

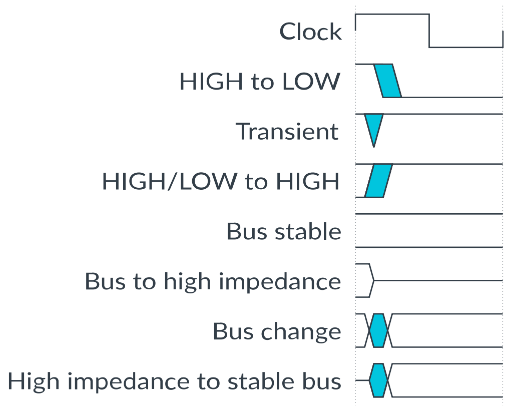
A reminder of something important that relates to the information you are reading.

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 1-1: 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, n denotes an active-LOW signal.

1.4 Additional reading

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

Table 1-2: Arm publications

Document name	Document ID	Licensee only
AMBA® APB Protocol Specification Version 2.0	IHI 0024C	No
AMBA® 4 ATB Protocol Specification ATBv1.0 and ATBv1.1	IHI 0032B	No
AMBA® Low Power Interface Specification, Arm® Q-Channel and P-Channel Interfaces	IHI 0068C	No
Arm® AMBA® 5 AHB Protocol Specification AHB5, AHB-Lite	IHI 0033B.b	No

Document name	Document ID	Licensee only
Arm® CoreLink™ LPD-500 Low Power Distributor Technical Reference Manual	100361	No
Arm® CoreSight™ Architecture Specification v3.0	IHI 0029E	No
Arm® CoreSight™ Base System Architecture 1.0 - Arm Platform Design Document	DEN 0068	No
Arm® CoreSight™ Program Flow Trace Architecture Specification PFTv1.0 and PFTv1.1	IHI 0035B	No
Arm® Cortex®-M3 Integration and Implementation Manual	DII 0240	Yes
Arm® Cortex®-M4 Integration and Implementation Manual	DII 0239	Yes
Arm® Debug Interface Architecture Specification ADIv6.0	IHI 0074B	No
Arm® Embedded Trace Macrocell Architecture Specification ETMv4.0 to ETMv4.3	IHI 0064E	No
Arm® Power Control System Architecture Specification Version 1.0	DEN 0050B	Yes

Table 1-3: Other publications

Organization	Document name
Accellera	*IP-XACT version 1685-2009*
IEEE	*IEEE 1149.1-2001 IEEE Standard Test Access Port and Boundary Scan Architecture (JTAG)*
IEEE	*Verilog-2001 Standard* IEEE Std 1364-2001

Note

Arm tests its PDFs only in Adobe Acrobat and Acrobat Reader. Arm cannot guarantee the quality of its documents when used with any other PDF reader.

Adobe PDF reader products can be downloaded at http://www.adobe.com

2. About the SoC-600M

This chapter introduces the CoreSight™ SoC-600M.

2.1 About this product

CoreSight™ SoC-600M is a member of the Arm embedded debug and trace component family.

Some of the features that CoreSight™ SoC-600M provides are:

- Components that can be used for debug and trace of Arm SoCs. These SoCs can be simple single-processor designs to complex multiprocessor and multi-cluster designs that include many heterogeneous processors.
- Support for the *Arm Debug Interface* (ADI) v6 and CoreSight™ v3 Architectures that enable you to build debug and trace functionality into your systems. It supports debug and trace over existing functional interfaces.
- Components that support the development of low-power system implementations through architected fine-grained power control.
- Q-Channel interfaces for clock and power quiescence.
- Can be integrated with the Arm® CoreLink™ LPD-500 as part of a full-chip power and clock control methodology.
- The Arm® CoreSight™ SDC-600 can be integrated with CoreSight™ SoC-600M, with an applicable licence, as part of a certificate-based authenticated debug solution.

The CoreSight™ SoC-600M bundle includes:

- A library of configurable CoreSight™ components that are written in Verilog, and that are compliant with the *Verilog-2001 Standard* IEEE Std 1364-2001.
- Example timing constraint files for each component in SDC format.

2.2 Features

Features and capabilities that the SoC-600M provides include:

Debug

- *Arm® Debug Interface Architecture Specification ADIv6.0*-compliant debug port. This debug port supports JTAG and Serial Wire protocols for connection to an off-chip debugger. This connection is achieved using a low-pin-count connection that is suitable for bare-metal debug and silicon bring-up.
- *Arm® CoreSight™ Architecture Specification v3.0* compliance enables debug over functional interfaces, suitable for application development and in-field debug without a dedicated debug interface.

- Infrastructure components supporting system identification and integration with other CoreSight™ IP.

Trace

- Versatile *Trace Memory Controller* (TMC) supporting local on-chip storage, and buffering of trace data.
- Infrastructure components supporting filtering and routing of trace data on chip.

Embedded Cross Triggering

- *Cross Trigger Interface* (CTI) supports up to 32 trigger inputs and outputs with a single component instance.
- *Cross Trigger Matrix* (CTM) supports up to 33 CTI or CTM connections without cascading.

Power

- *Arm® CoreSight™ Architecture Specification v3.0*-compliant *Granular Power Requester* (GPR) enables fine-grained debug and system power control at all levels of debug hierarchy.
- Components are designed for low-power implementation, supporting clock and power quiescence and wakeup signaling where necessary.
- Components support Q-Channel *Low-Power Interfaces* (LPI) for integration with power controllers to support system-level clock and power gating where necessary.
- Infrastructure components support implementation across multiple clock and power domains.

Miscellaneous

- Some components, such as the bridges and *Serial Wire Debug Port* (SW-DP), use two Verilog modules to span clock and power domains. This design can ease implementation in complex SoC designs that have multiple clock and power domains.
- Infrastructure components support integration with legacy IP including *Arm® CoreSight™ Architecture Specification v2.0*-compliant, and JTAG components.

2.3 Supported standards

CoreSight™ SoC-600M is compliant with the following standards.

- *Arm® CoreSight™ Architecture Specification v3.0*
- *AMBA® APB Protocol Specification Version 2.0*
- *Arm® AMBA® 4 ATB Protocol Specification ATBv1.0 and ATBv1.1*
- *Arm® Debug Interface Architecture Specification ADIv6.0*
- *Arm® AMBA® 5 AHB Protocol Specification AHB5, AHB-Lite*
- *AMBA® Low Power Interface Specification, Arm® Q-Channel and P-Channel Interfaces*
- *Arm® CoreSight™ Program Flow Trace Architecture Specification PFTv1.0 and PFTv1.1*
- *Verilog-2001 Standard* IEEE Std 1364-2001

- Accellera, *IP-XACT version 1685-2009*
- *IEEE 1149.1-2001 IEEE Standard Test Access Port and Boundary Scan Architecture (JTAG)*

2.4 Documentation

The SoC-600M documentation includes a *Technical Reference Manual* (TRM) and a *Configuration and Integration Manual* (CIM). These books relate to the SoC-600M design flow.

Technical Reference Manual

The TRM describes the functionality and the effects of functional options on the behavior of the SoC-600M components. It is required at all stages of the design flow. The choices that you make in the design flow can mean that some behavior that is described in the TRM is not relevant. If you are programming a device that is based on SoC-600M components, then contact the integrator to determine the configuration of your device.

Configuration and Integration Manual

The CIM describes:

- How to configure the SoC-600M components
- How to integrate the SoC-600M components into your SoC design and how to configure system-specific Identification Registers
- How to implement the SoC-600M components to produce a hard macrocell of the design. This description includes custom cell replacement, a description of the power domains, and a description of the design synthesis.

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

2.5 Design process

The SoC-600M components are delivered as synthesizable Verilog RTL.

Before the SoC-600M components can be used in a product, they must go through the following processes:

System design

Determining the necessary structure and interconnections of the SoC-600M components that form the CoreSight™ debug and trace subsystem.

Configuration

Defining the memory map of the system and the functional configuration of the SoC-600M components.

Integration

Connecting the SoC-600M components together, and to the SoC memory system and peripherals.

Verification

Verifying that the CoreSight™ debug and trace subsystem 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™ debug and trace subsystem by programming specific values into registers that affect the behavior of the SoC-600M components.

Note

SoC-600M is highly configurable to support many system topologies. Arm recommends that you follow the guidance in the *Arm® CoreSight Base System Architecture - Arm Platform Design Document*. This ensures wide support for your product across the Arm debug ecosystem.

2.6 Component list

CoreSight™ SoC-600M components are provided as RTL blocks, the name of each one prefixed with `css600_`.

The following table shows the components and their versions. The Revision column only applies to components with a programmers model. The Revision value is for the PIDR2.REVISION field.

Table 2-1: SoC-600M component list

Name	Description	Version	Revision	IP-XACT Version
`css600_ahbap`	AHB Access Port	r2p1	4	r2p1_1
`css600_apb3toapb4adapter`	APB3 to APB4 adapter	r0p0	-	r0p0_1
`css600_apb4toapb3adapter`	APB4 to APB3 adapter	r0p0	-	r0p0_1
`css600_apbap`	APB Access Port	r1p1	3	r1p1_1
`css600_apbasyncbridge`	APB Asynchronous Bridge	r0p3	-	r0p3_0
`css600_apbic`	APB Interconnect	r0p2	-	r0p2_1
`css600_apbpaddrdbg31adapter`	APB PADDRDBG[31] Adapter	r1p0	-	r1p0_0
`css600_apbrom`	APB ROM table	r0p1	-	r0p1_2
`css600_apbsyncbridge`	APB Synchronous Bridge	r0p3	-	r0p3_0
`css600_apv1adapter`	Access Port v1 Adapter	r0p0	0	r0p0_1
`css600_atbasyncbridge`	ATB Asynchronous Bridge	r1p1	-	r1p1_0

Name	Description	Version	Revision	IP-XACT Version
css600_atbbuffer	ATB Trace Buffer	r0p2	-	r0p2_0
css600_atbdownsizer	ATB Downsizer	r0p1	-	r0p1_0
css600_atbfunnel	ATB Trace Funnel	r0p3	3	r0p3_1
css600_atbreplicator	ATB Trace Replicator	r0p3	3	r0p3_1
css600_atbsyncbridge	ATB Synchronous Bridge	r1p3	-	r1p3_0
css600_atbupsizer	ATB Trace Upsizer	r0p2	-	r0p2_0
css600_authasyncbridge	Authentication Asynchronous Bridge	r0p0	-	r0p0_0
css600_authreplicator	Authentication Replicator	r0p0	-	r0p0_0
css600_authsyncbridge	Authentication Synchronous Bridge	r0p0	-	r0p0_0
css600_channelpulseasyncbridge	Channel Pulse Asynchronous Bridge	r0p2	-	r0p2_0
css600_channelpulsesyncbridge	Channel Pulse Synchronous Bridge	r0p2	-	r0p2_0
css600_channelpulsetochanneladapter	Channel Pulse to Channel Adapter	r0p1	-	r0p1_0
css600_channeltochannelpulseadapter	Channel to Channel Pulse Adapter	r0p2	-	r0p2_0
css600_cortexm0integrationcs	Cortex-M0 PIL	r0p0	1	r0p0_0
css600_cortexm3integrationcs	Cortex-M3 PIL	r0p0	1	r0p0_0
css600_cortexm4integrationcs	Cortex-M4 PIL	r0p0	1	r0p0_0
css600_cti	Cross Trigger Interface	r0p3	3	r0p3_1
css600_ctitostmadapter	CTI to STM Adapter	r0p0	-	r0p0_0
css600_ctm	Cross Trigger Matrix	r0p0	-	r0p0_0
css600_dp	Debug Port	r0p4	4	r0p4_1
css600_dpabortasyncbridge	DP Abort Asynchronous Bridge	r0p2	-	r0p2_0
css600_dpabortreplicator	DP Abort Replicator	r0p0	-	r0p0_0
css600_dpabortsyncbridge	DP Abort Synchronous Bridge	r0p2	-	r0p2_0
css600_eventlevelasyncbridge	Event Level Asynchronous Bridge	r0p0	-	r0p0_0
css600_eventlevelsyncbridge	Event Level Synchronous Bridge	r0p0	-	r0p0_0
css600_eventpulseasyncbridge	Event Pulse Asynchronous Bridge	r0p2	-	r0p2_0
css600_eventpulsesyncbridge	Event Pulse Synchronous Bridge	r0p2	-	r0p2_0
css600_eventpulsetoeventadapter	Event Pulse to Event Adapter	r0p1	-	r0p1_0
css600_eventtoeventpulseadapter	Event to Event Pulse Adapter	r0p2	-	r0p2_0
css600_jtagap	JTAG Access Port	r0p3	3	r0p3_1
css600_jtagtoswjadapter	JTAG to SWJ Adapter	r0p0	-	r0p0_0
css600_swjic	SWJ Interconnect	r1p0	-	r1p0_0
css600_swjtojtagadapter	SWJ to JTAG Adapter	r0p0	-	r0p0_0
css600_tmc	Trace Memory Controller	r0p6	6	r0p6_2
css600_tpiu	Traceport Interface Unit	r1p1	2	r1p1_1
css600_tsgen	Timestamp Generator	r0p0	0	r0p0_2
css600_tsintp	Timestamp Interpolator	r0p2	-	r0p2_0
css600_tsreplicator	Timestamp Replicator	r0p0	-	r0p0_0

2.7 Product revisions

This section describes the differences in functionality between product revisions of the CoreSight™ SoC-600M.

r0p0

First release of CoreSight™ SoC-600M.

r1p0

First release at REL quality. Component errata fixes. New AHB-AP major revision. This updategives the debugger greater control over AHB transaction attributes that is required for the latestArm Cortex-M processors. This change requires an update to the programmer's model.

r1p1

Second release at REL quality. Component errata fixes.

3. DAP components functional description

This chapter describes the functionality of the SoC-600M.

3.1 Debug port

The `css600_dp` module implements the JTAG and Serial Wire Debug Port protocols. Either of these protocols can be omitted to save area in systems that do not require both protocols.

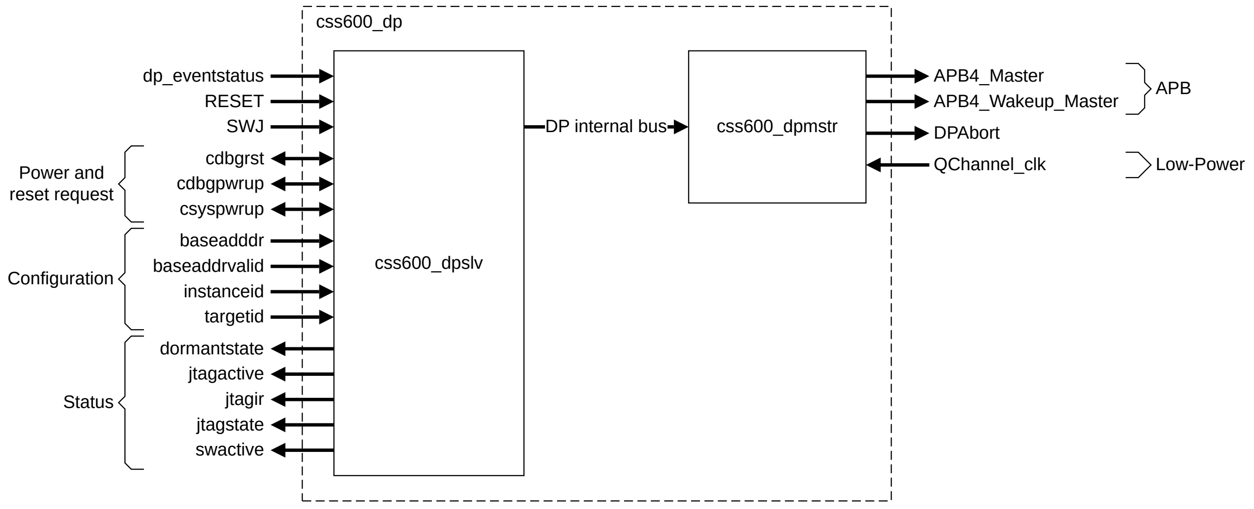
The debug port communicates with the debug components through the APB infrastructure that is connected to the debug port APB master interface.

The debug port implements the following features:

- ADIv6 architecture
- Single clock domain in each part
- Asynchronous bridge between the slave and master parts
- 4-bit or 8-bit Instruction register for JTAG implementation
- Separate slave and master components, implementing JTAG, Serial Wire, or both in the slave, and APB in the master

The following figure shows the external connections on the *Debug Port* (DP).

Figure 3-1: `css600_dp` logical connections



3.2 Memory Access Ports

Memory Access Ports connect one memory system to another using one of the AMBA bus protocols: AHB or APB.

The *Arm® Debug Interface Architecture Specification ADIv6.0* defines a *Memory Access Port* (MEM-AP) so that it provides two logical views of the access port to the debugger. These two views are referred to as twin APs or logical APs. In SoC-600M, these two logical APs are contiguous in the memory map and each one of them occupies 4kB address space. An external debugger can only discover one of the twin APs through the ROM table. The other AP is dedicated for self-hosted debug. The MEM-AP itself is not capable of differentiating which of the twin APs is visible in the ROM table. The MEM-AP decodes the access requests on the APB slave interface and maps them to AP-L0 or AP-L1, based on the value of paddr_s[12].

The following table shows the implementation-defined features of MEM APs that SoC-600M supports.

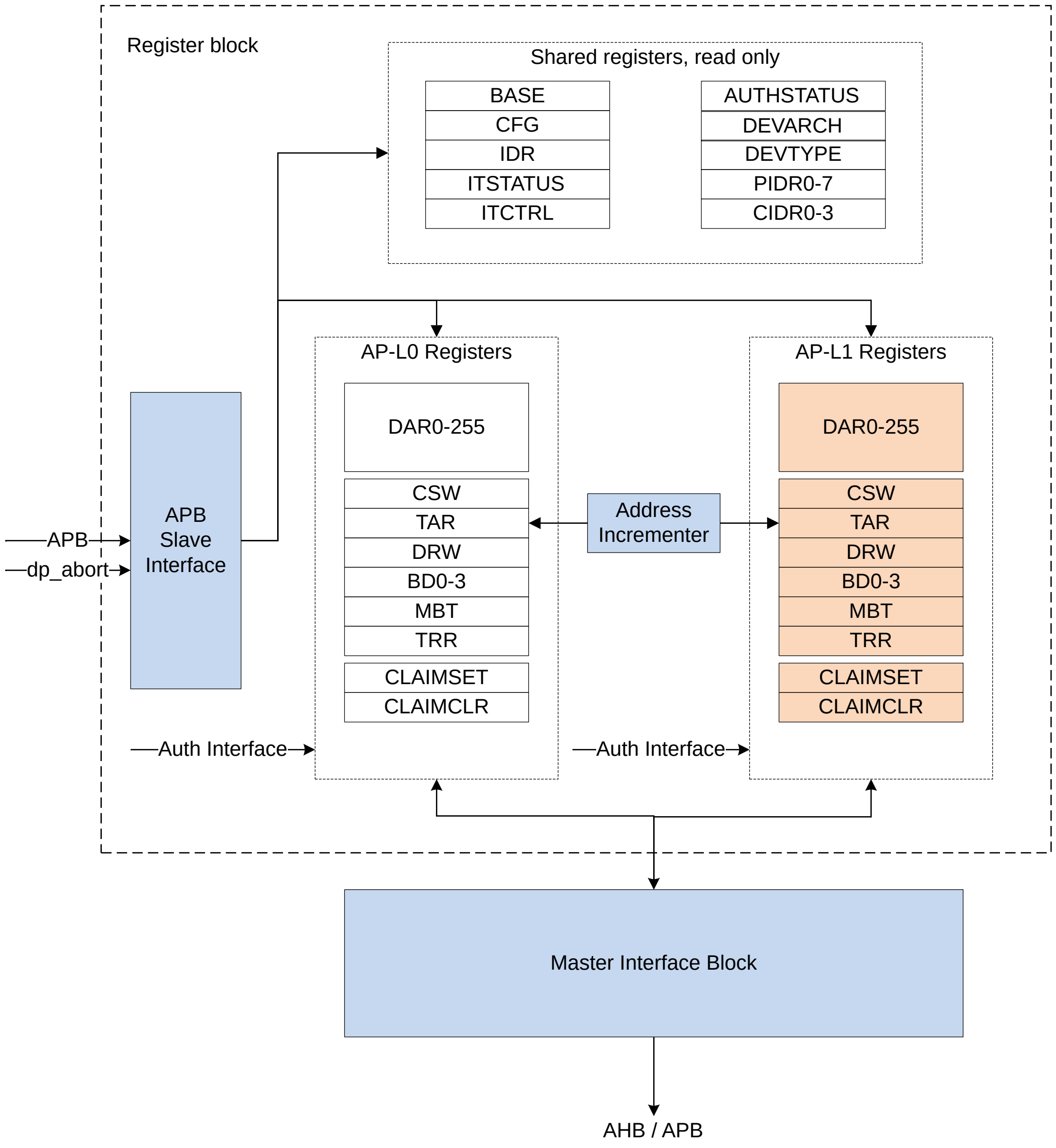
Table 3-1: MEM-AP implementation-defined features

Feature	AHB-AP	APB-AP
Packed transfers	No	No
Non-word sizes (smaller than 32-bit)	Supported	No
Large Data Extension (64-bit)	No	No
Large Physical Address Extension (64-bit)	No	No
Barrier Operation Extension	No	No
Memory Tagging Extension, MTE	No	No

The slave and master interfaces are shared by both, AP-L0 and AP-L1. They also share all read-only registers, as the following figure shows, so that the read value returned by these registers is the same, irrespective of whether they are accessed through AP-L0 or AP-L1. However, the writeable registers are duplicated on both logical APs and accessing them in one view does not affect the state in the other view.

The following diagram shows the MEM AP block diagram.

Figure 3-2: MEM-AP block diagram



3.2.1 APB Access Port

The `css600_apbap` module is a *Memory Access Port* (MEM-AP). The `css600_apbap` is an APB4 slave component that provides access to another APB4 memory system.

Use the `css600_apbap` to provide access to an APB4 memory space, for example:

- A subsystem of CoreSight™ components that includes Arm® Cortex®-A or Cortex®-R processors

- A subsystem of CoreSight™ components
- Any other APB4 memory system

The APB Access Port allows visibility into another memory system from the debug APB infrastructure. Access Ports and related infrastructure can be cascaded in a CoreSight™ system to any depth. This process allows any memory system to contain a window into another memory system with a maximum memory footprint of 8KB in the source memory system.

The APB-AP provides an AMBA APB4 slave interface for programming and an AMBA APB4 master interface for accessing the target memory system. The programmers model contains the details of the registers for accessing the features of the APB4 master interface.

The APB-AP provides the following features:

- Error response
- Stalling accesses
- Little-endian only
- Single clock domain
- 32 bits data access only
- Auto-incrementing *Transfer Address Register* (TAR)
- An APB4 slave interface
- An APB4 master interface
- An Access Port Enable interface
- CoreSight Component base pointer register
- A Q-Channel LPI for high-level clock management

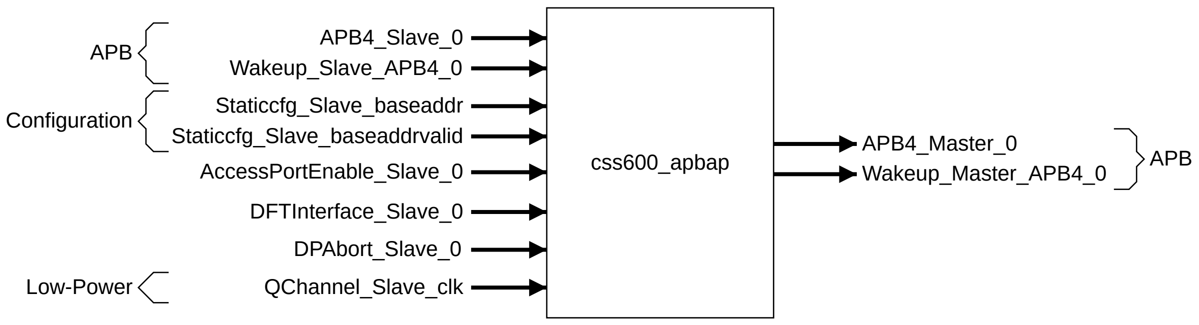
The APB-AP does not support subword transfers.

Note

If the DP issues an abort over the Debug APB interface, the APB-AP completes the transaction on its Debug APB slave interface immediately. The DAP transfer abort does not cancel the ongoing APB transfer on the APB master interface.

The following figure shows the external connections on the APB Access Port.

Figure 3-3: `css600_apbap` logical connections



3.2.2 AHB Access Port

The `css600_ahbap` module is a *Memory Access Port* (MEM-AP). The `css600_ahbap` is an APB4 slave component that provides access to an AHB5 memory system.

Use the `css600_ahbap` to provide access to an AHB5 memory space, for example:

- An Arm® Cortex®-M processor and subsystem
- Any other AHB5 memory system

The AHB Access Port allows visibility into another memory system from the debug APB infrastructure. Access Ports and related infrastructure can be cascaded in a CoreSight™ system to any depth. This process allows any memory system to contain a window into another memory system with a maximum memory footprint of 8KB in the source memory system.

The AHB-AP provides an AMBA® APB4 slave interface for programming and an AMBA® AHB5 master interface for accessing the target memory system. The programmers model contains the details of the registers for accessing the features of the AHB master interface.

The AHB-AP provides the following features:

- Error response
- Stalling accesses
- Little-endian only
- Single clock domain
- Auto-incrementing *Transfer Address Register* (TAR)
- An APB4 slave interface
- An AHB5 master interface
- An Access Port Enable interface
- 8 bits, 16 bits, or 32 bits data access
- CoreSight Component base pointer register
- Support for AHB5 TrustZone® signaling

- A Q-Channel LPI for high-level clock management

The AHB-AP does not support:

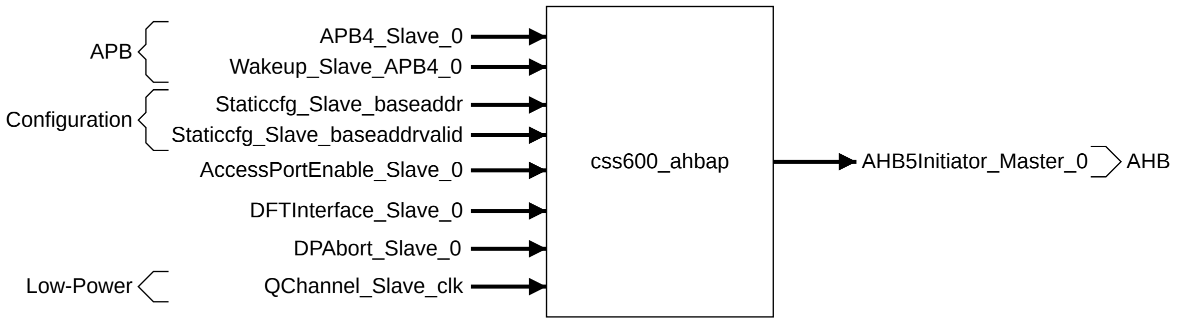
- Exclusive accesses
- Unaligned transfers
- BURST or SEQ transactions

Note

If the DP issues an abort to AHB-AP, the AHB-AP completes the transaction on its APB slave interface immediately. The DAP transfer abort does not cancel the ongoing AHB transfer.

The following figure shows the external connections on the AHB Access Port.

Figure 3-4: `css600_ahbap` logical connections



3.2.3 Error response handling

CoreSight™ SoC-600M *Memory Access Ports* (MEM-APs) implement Error Response Handling Version 1.

Error Response Handling V1 is defined in the *Arm® Debug Interface Architecture Specification ADIv6.0*. Support for this error handling mechanism is indicated in the CFG.ERR register field. The three register bits CSW.ERRNPASS, CSW.ERRSTOP, and TRR.ERR are used to define the behavior of this feature. For more information, see the relevant programmers model register descriptions.

The MEM-AP logs errors in Transfer Response Register by setting TRR.ERR bit to 1. When set, this bit remains set until software clears it by writing 1 to it. The following types of memory access errors are logged:

Access Port Enable failure

This error is caused by an unauthenticated memory access attempt, such as:

- Any memory access when ap_en is LOW.
- A Secure memory access when ap_secure_en is LOW.

Stopped on error	This error is due to a memory access attempt when TRR.ERR=1 and CSW.ERRSTOP=1.
AHB/APB error	An error response that is received on the AP master interface indicating that the memory access failed.
Abort	Aborted memory transfers.
Master busy	This error happens if a memory access is attempted after an abort, but while the CSW.TrInProg bit is still set.

Internal register access errors are not logged in the TRR but are always passed on the APB slave interface. If a register write is attempted after an abort while the CSW.TrInProg bit is set, an error is generated.

The register bit CSW.ERRNPASS controls whether a memory access error is passed back to the requestor. The internal register access errors are always passed back on the APB slave interface regardless of the value of this bit.

The CSW.ERRNPASS bit has the following effect on behavior:

0	Memory access errors are passed back on the APB slave interface.
1	Memory access errors are not passed back on the APB slave interface. In this case, a normal APB response is returned even for failed memory transactions.

There are two to this rule. In each case, the error is always passed on the APB slave interface, regardless of the status of the CSW.ERRNPASS bit. The exceptions are:

- If the memory transaction is aborted.
- If the error is generated by a memory access attempt, while the CSW.TrInProg bit is still set from a previously aborted access.

The APB read data for all transactions that generate an error is **UNKNOWN**.

If no previous memory access errors are logged, that is TRR.ERR=0, memory accesses are allowed, regardless of the state of CSW.ERRSTOP.

If a previous memory access error is still logged, that is TRR.ERR=1, the register field CSW.ERRSTOP controls whether to prevent memory accesses as follows:

0	New memory accesses are allowed.
1	No new memory accesses are allowed, and any new memory accesses result in an error response on the APB slave interface, provided CSW.ERRNPASS is 0. In this case, TRR.ERR remains set and the memory transfer is not initiated.

The following table summarizes this MEM-AP behavior for memory errors other than Abort and Master Busy.

Table 3-2: MEM-AP behavior for memory errors other than Abort and Master Busy

TRR.ERR	CSW.ERRNPASS	CSW.ERRSTOP	New memory access	Slave error	Error logged
0	0	x	Allowed if Authenticated by the Access Port Enable interface, otherwise blocked.	Passed	Yes
0	1	x		Not passed	Yes
1	0	0		Passed	Yes
1	1	0		Not passed	Yes
1	0	1	Blocked	Passed	Yes
1	1	1	Blocked	Not passed	Yes

The twin logical APs implement error handling independently, and the errors that are received or generated on one do not affect the other.

Memory errors, other than Abort and Master-Busy, are maskable errors. That is, they can be masked from appearing on an APB slave interface by setting the CSW.ERRNPASS bit. It is possible for a single memory access to cause multiple error sources to generate errors at the same time. For example, a memory access can trigger a stop-on-error and an authentication failure.

If an error is masked, an error response is passed on the APB slave interface, even if CSW.ERRNPASS is 1, and if at least one of the sources of error is non-maskable (Abort or Master-Busy). If all the triggered error sources are maskable, the error is passed only if CSW.ERRNPASS is 0.

If the Access Port Enable interface signals change while a memory transfer is in progress, the MEM-AP still completes the ongoing transfer normally. The new Access Port Enable interface values then take effect from the next transaction. If a memory access request is received while the MEM-AP is in Q_STOPPED state, the authentication signal values are sampled only after entering Q_RUN state in the first cycle, and that value is used to determine whether to allow or block the pending APB transfer.

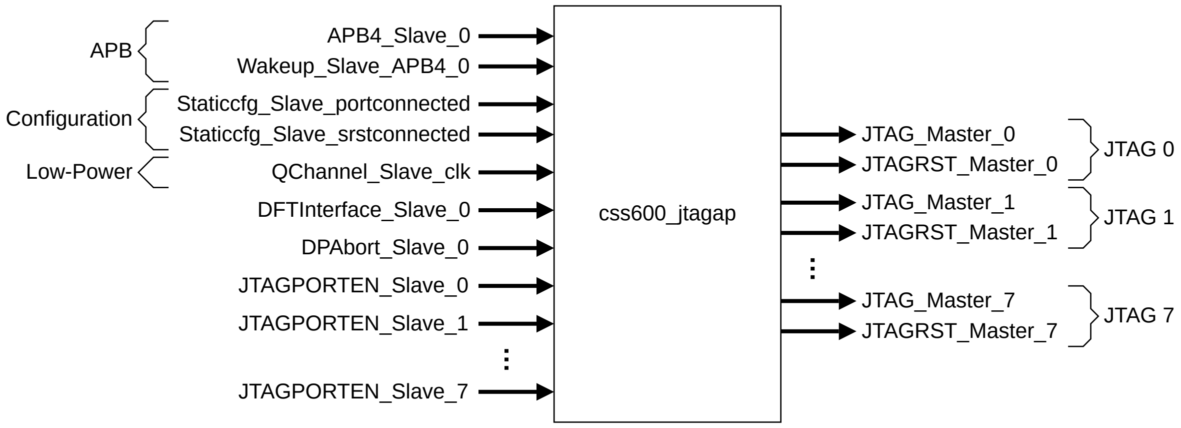
3.3 JTAG Access Port

The `css600_jtagap` provides JTAG access to on-chip components, operating as a JTAG master port to drive JTAG chains throughout a SoC.

The JTAG command protocol is byte-oriented, with a word wrapper on the read and write ports to yield acceptable performance from the 32-bit internal data bus in the DAP. Daisy chaining is avoided by using a port multiplexer. These two features prevent slower cores from impeding faster cores.

The following figure shows the external connections on the JTAG Access Port.

Figure 3-5: `css600_jtagap` logical connections



For more information, see the *Arm® Debug Interface Architecture Specification ADIv6.0*.

3.4 Access Port v1 adapter

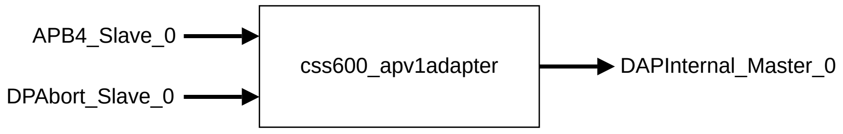
Use the `css600_apv1adapter` to connect a legacy *Access Port* (AP) with a *DAP Internal* (DAPBus) slave interface into a CoreSight™ Architecture v3 system.

The `css600_apv1adapter`:

- Maps the legacy AP registers into the *Arm® CoreSight™ Architecture Specification v3.0* APB4 memory map
- Provides ID registers that allow a debugger to identify the combination as a mapped legacy Access Port
- Provides integration registers that allow a debugger to check connectivity of the *DP Abort* signal

The following figure shows the external connections on the Access Port v1 Adapter.

Figure 3-6: `css600_apv1adapter` logical connections



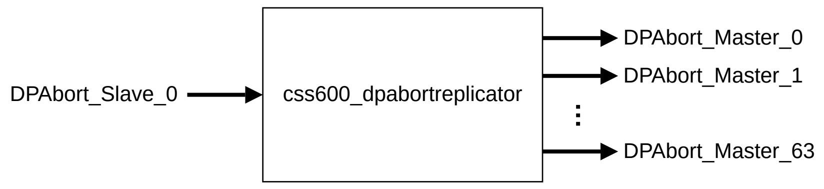
3.5 DP Abort replicator

The `css600_dpabortreplicator` is an IP-XACT *phantom component* that is provided to support stitching in an IP-XACT tooling product. There is no Verilog module for `css600_dpabortreplicator`.

Use the `css600_dpabortreplicator` to connect a single DP Abort master interface to multiple DP Abort slave interfaces. You must connect the DP Abort output from the Debug Port to every Access Port that appears in the Debug Port memory space.

The following figure shows the external connections on the DP Abort Replicator.

Figure 3-7: `css600_dpabortreplicator` logical connections



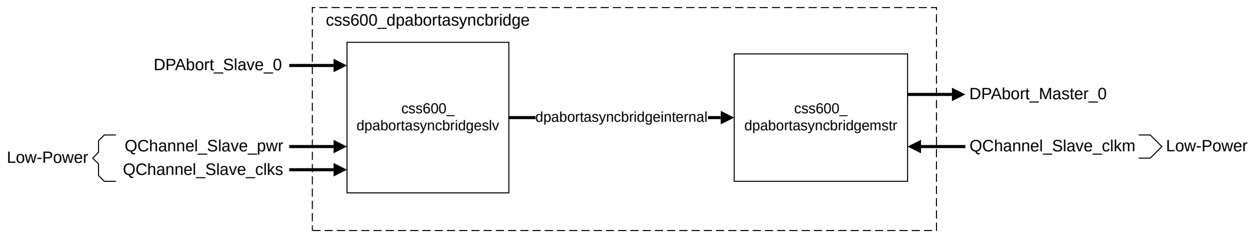
3.6 DP Abort asynchronous bridge

The `css600_dpabortasyncbridge` is a wrapper component that instantiates a pulse asynchronous bridge.

The bridge is used to transfer the dp_abort signal across a clock or power domain boundary. The dp_abort signal is a pulse event that is used to unlock a deadlocked transaction on a DP to AP interconnection.

The following figure shows the external connections on the DP Abort asynchronous bridge.

Figure 3-8: `css600_dpabortasyncbridge` logical connections


css600_dpabortasyncbridge
DPAbort_Slave_0
Low-Power
QChannel_Slave_pwr
QChannel_Slave_clks
css600_ dpabortasyncbridgeslv
dpabortasyncbridgeinternal
css600_ dpabortasyncbridgemstr
DPAbort_Master_0
QChannel_Slave_clkm
Low-Power

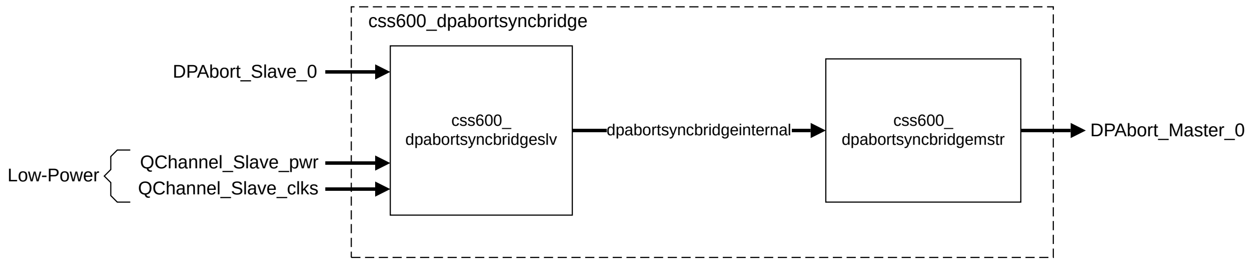
3.7 DP Abort synchronous bridge

The `css600_dpabortsyncbridge` is a wrapper component that instantiates a pulse synchronous bridge.

The bridge is used to transfer the dp_abort signal across a power domain boundary. The dp_abort signal is a pulse event that is used to unlock a deadlocked transaction on a DP to AP interconnection.

The following figure shows the external connections on the DP Abort synchronous bridge.

Figure 3-9: `css600_dpabortsyncbridge` logical connections



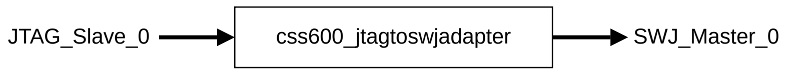
3.8 JTAG to SWJ adapter

The `css600_jtagtoswjadapter` is an IP-XACT phantom component that is provided to support stitching in an IP-XACT tooling product. There is no Verilog module for `css600_jtagtoswjadapter`.

Use the `css600_jtagtoswjadapter` to connect a JTAG master interface to a *Serial Wire/JTAG* (SWJ) slave interface. This might be necessary when connecting a Debug Port to the `css600_jtagap`.

The following figure shows the external connections on the JTAG to SWJ adapter.

Figure 3-10: `css600_jtagtoswjadapter` logical connections



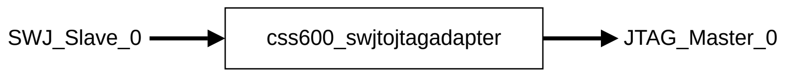
3.9 SWJ to JTAG adapter

The `css600_swjtojtagadapter` is provided to support stitching in an IP-XACT tooling product.

Use the `css600_swjtojtagadapter` to connect a *Serial Wire/JTAG* (SWJ) master interface to a JTAG slave interface.

The following figure shows the external connections on the SWJ to JTAG adapter.

Figure 3-11: `css600_swjtojtagadapter` logical connections



3.10 SWJ interconnect

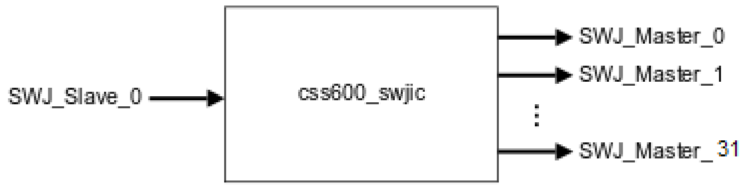
The `css600_swjic` is a Serial Wire and JTAG interconnect that enables you to connect multiple SWJ slave components, for example Debug Ports, to a single SWJ master.

Use the `css600_swjic` to:

- Daisy-chain multiple JTAG Debug Ports
- Combine the data and control signals from multiple Serial Wire Multi Drop Debug Ports

The following figure shows the external connections on the SWJIC.

Figure 3-12: `css600_swjic` logical connections



Note

Creating long JTAG scan chains can create performance issues. Arm recommends that you avoid creating long daisy-chains if possible.

4. APB infrastructure components functional description

This chapter describes the functionality of the APB infrastructure components.

4.1 APB interconnect

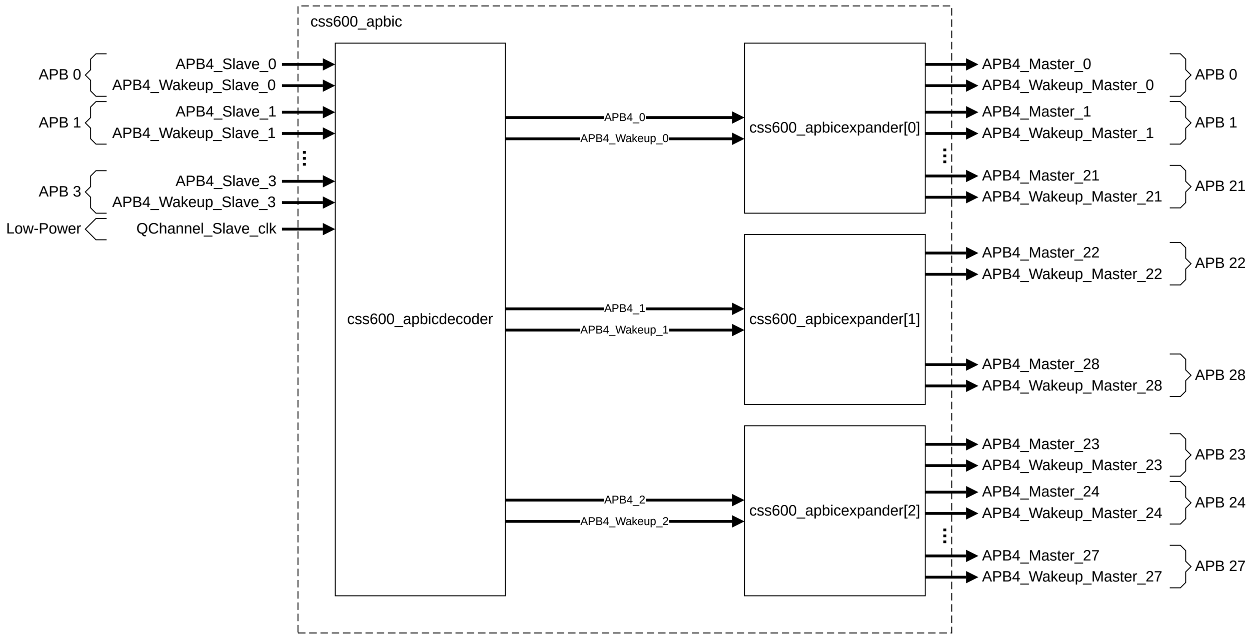
The `css600_apbic` is used to provide connections between APB4 masters and APB4 slaves anywhere in a CoreSight system.

APB4 masters can be debug ports, APB Access Ports, or other APB masters from a compute subsystem. It is a two-part meta-component that has the following features:

- Single clock domain
- Decoder component configurable for up to four slave interfaces and up to 64 master interfaces
- Physical grouping of decoded master interfaces using one or more configurable expander components
- Option to insert APB asynchronous or synchronous bridges between decoder and expander instances to cross power and clock domain boundaries
- Configurable APB address widths to suit addressable ranges
- A Q-Channel LPI for high-level clock management

The following figure shows the external connections on the APB interconnect.

Figure 4-1: `css600_apbic` logical connections



4.1.1 Arbitration

The internal arbiter arbitrates between competing slave interfaces for access to the debug APB.

When a slave interface raises a request, the arbiter gives the highest priority to the slave interface with the lowest instance suffix. For example, Slave Interface 0 > Slave Interface 1 > Slave Interface 2 > Slave Interface 3. The order in which the slave interfaces raised their requests relative to each other is not used in arbitration.

The arbitration is re-evaluated after every access.

4.1.2 Error response

The APB interconnect returns an error on its slave interface under certain conditions.

An error response is returned when either:

- The targeted APB slave returns an error response
- A slave interface accesses an address that does not decode to any connected APB slave

4.2 APB ROM table

The `css600_apbrom` module is a *ROM table* with an APB4 slave interface.

The `css600_apbrom_gpr` is a *ROM Table* that includes the *Granular Power Requestor* (GPR) function.

Use the `css600_apbrom` or `css600_apbrom_gpr` to:

- Identify part of your system or subsystem.
- Indicate the locations of other CoreSight™ components in the same address space to an External Debugger.
- Request power or reset to be supplied to components in the debug subsystem or the wider system (`css600_apbrom_gpr` only).

The `css600_apbrom` and `css600_apbrom_gpr` support up to 512 32-bit component entries, which are set by configuration parameters. The `css600_apbrom` and `css600_apbrom_gpr` support dynamic control of the *ROM table* IDs and, optionally the presence of each entry, using configuration input signals. These features make the *ROM table* suitable for use in configurable and hardened subsystems.

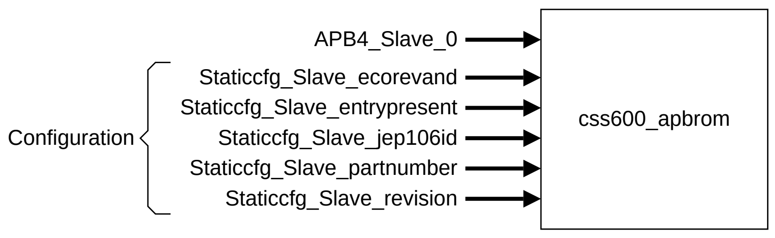
The `css600_apbrom_gpr` version adds the capability to request power or reset to individual components or parts of a system through a power or reset controller that is implemented outside the CoreSight™ subsystem. Power request interface numbers are normally aligned to power domain IDs configured into the *ROM table*.

The GPR configuration provides the following additional features:

- Authentication interface to control access to power and reset control features
- Configurable number, up to 32, of debug power request interfaces, comprising a cdbgpwrupreq and cdbgpwrupack pair of signals
- Configurable number, up to 32, of system power request interfaces, comprising a csyspwrupreq and csyspwrupack pair of signals
- A debug reset request interface, comprising a cdbgrstreq and cdbgrstack pair of signals
- A system reset request interface, comprising a csysrstreq and csysrstack pair of signals

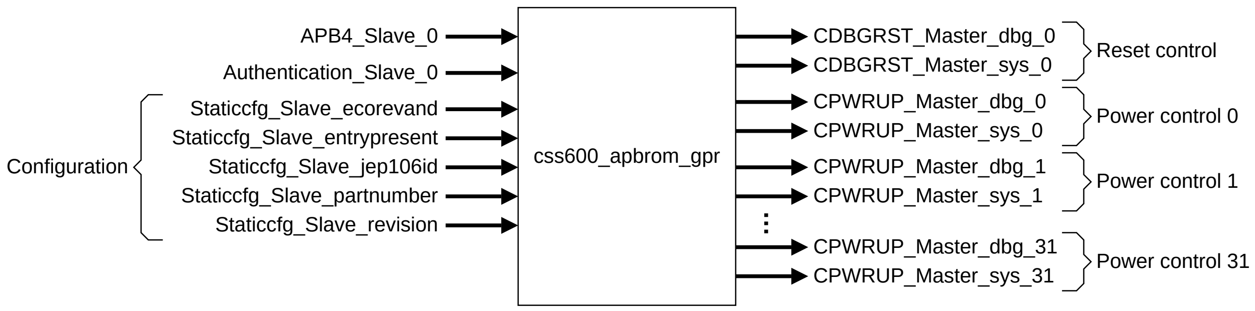
The following figure shows the external connections on the APB ROM table.

Figure 4-2: `css600_apbrom` logical connections



The following figure shows the external connections on the APB ROM table with GPR.

Figure 4-3: `css600_apbrom_gpr` logical connections



4.3 APB asynchronous bridge

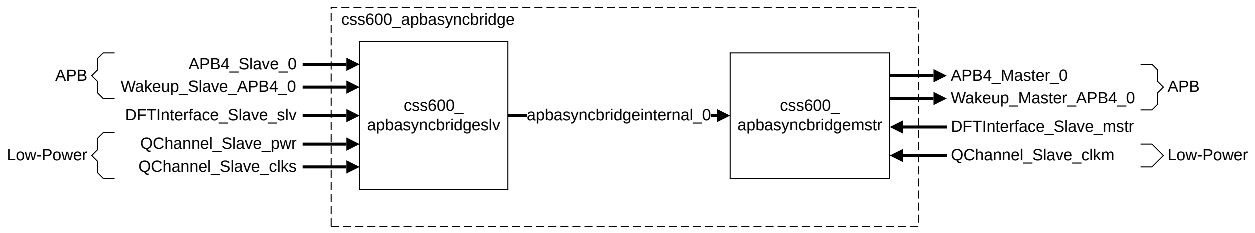
The `css600_apbasyncbridge` is used where an AMBA APB4 bus is required to cross a clock or power domain boundary.

The APB asynchronous bridge provides the following features:

- Two independent clock domains with any phase or frequency alignment
- Two independent power domains, either of which can be switched relative to the other
- Three Q-Channel LPIs for slave side clock, master side clock, and power switching management
- A two-part meta-component with separate slave and master side components
- Configurable APB address width
- Configurable 2- or 3-deep synchronizers

The following figure shows the external connections on the APB asynchronous bridge.

Figure 4-4: `css600_apbasyncbridge` logical connections



4.4 APB synchronous bridge

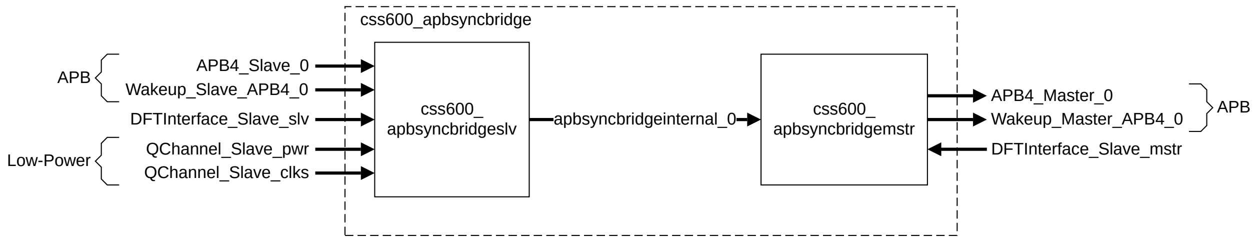
The `css600_apbsyncbridge` is used where an AMBA APB4 bus is required to cross a clock domain boundary between two synchronous clocks.

The APB asynchronous bridge provides the following features:

- Two synchronous clock domains with any frequency difference. The clocks must be skew balanced and from a common source, so that they are high-level-gated by a common control point.
- Two Q-Channel LPIs for clock and power switching management
- Two-part meta-component with separate slave and master side components
- Configurable APB address width
- Configurable 2-deep or 3-deep synchronizers for Q-Channel inputs

The following figure shows the external connections on the APB synchronous bridge.

Figure 4-5: `css600_apbsyncbridge` logical connections



4.5 APB PADDRDBG31 adapter

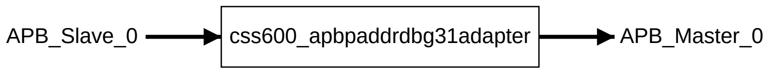
The `css600_apbpaddrdbg31adapter` module, enables you to integrate a CoreSight™ Architecture v2.0 component or subsystem into a CoreSight™ debug and trace subsystem (CSSYS).

Use the `css600_apbpaddrdbg31adapter` to integrate legacy components that have a dedicated paddrdbg31 signal into the memory map of the Arm® CoreSight™ SoC-600M CSSYS. The `css600_apbpaddrdbg31adapter`:

- Replaces the 2GB split at `0x80000000` with a user-defined split
- Maps the two views of the component to consecutive regions of the memory map
- Maps the external debugger view to the lower region
- Maps the self-hosted view to the upper region

The following figure shows the external connections on the APB PADDRDBG31 Adapter.

Figure 4-6: `css600_apbpaddrdbg31adapter` logical connections



4.6 APB3 to APB4 adapter

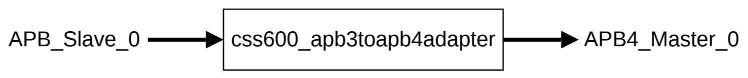
The `css600_apb3toapb4adapter` is an IP-XACT phantom component that is provided to support stitching in an IP-XACT tooling product.

There is no Verilog module for `css600_apb3toapb4adapter`.

Use the `css600_apb3toapb4adapter` to connect an APB3 master to an APB4 slave interface.

The following figure shows the external connections on the APB3 to APB4 adapter.

Figure 4-7: `css600_apb3toapb4adapter` logical connections



4.7 APB4 to APB3 adapter

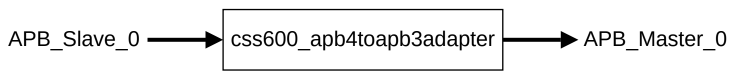
The `css600_apb4toapb3adapter` is an IP-XACT phantom component that is provided to support stitching in an IP-XACT tooling product.

There is no Verilog module for `css600_apb4toapb3adapter`.

Use the `css600_apb4toapb3adapter` to connect an APB4 master to an APB3 slave interface.

The following figure shows the external connections on the APB4 to APB3 adapter.

Figure 4-8: `css600_apb4toapb3adapter` logical connections



5. AMBA Trace Bus infrastructure components functional description

This chapter describes the functionality of the *AMBA Trace Bus* (ATB) infrastructure components.

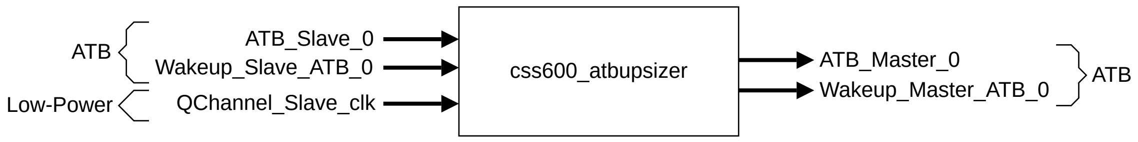
5.1 ATB upsizer

The `css600_atbupsizer` module enables you to increase the data width of an *AMBA Trace Bus* (ATB).

Use the `css600_atbupsizer` when you connect an AMBA® Trace Bus master interface to a wider AMBA® Trace Bus slave interface.

The following figure shows the external connections on the ATB upsizer.

Figure 5-1: `css600_atbupsizer` logical connections



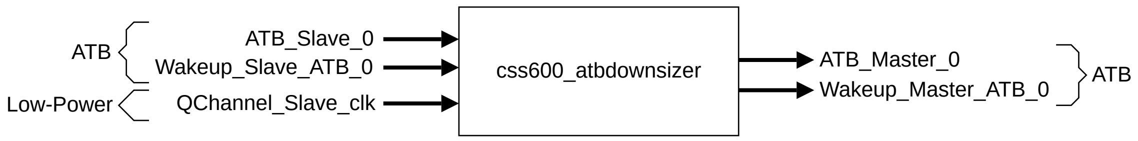
5.2 ATB downsizer

The `css600_atbdownsizer` module enables you to reduce the data width of an AMBA® Trace Bus.

Use the `css600_atbdownsizer` when you must connect an AMBA® Trace Bus master interface to a narrower AMBA® Trace Bus slave interface.

The following figure shows the external connections on the ATB downsizer.

Figure 5-2: `css600_atbdownsizer` logical connections



5.3 ATB funnel

The `css600_atbfunnel` is used when more than one trace source must be merged into a single trace stream.

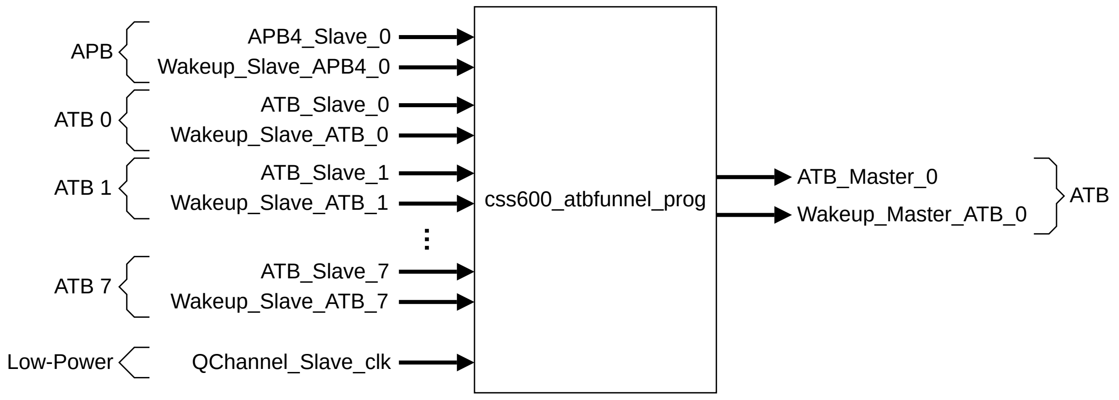
The funnel is configurable for the number of slave interfaces, from 2-8, and comes in programmable or non-programmable configurations. The programmers model section describes the register map of the programmable version.

The programmable configuration allows the following features:

- Independent enable control for each slave port
- Independent priority setting for each slave port, so that higher priority ports are serviced ahead of lower priority ports
- Programmable hold time to reduce input switching that is based on trace ID value
- Registers to allow integration testing of the trace network

The following figure shows the external connections on the programmable ATB funnel.

Figure 5-3: `css600_atbfunnel_prog` logical connections

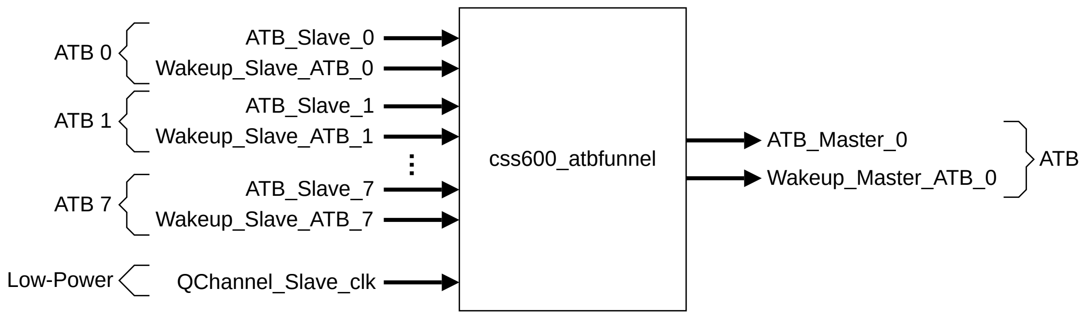


The non-programmable configuration has the following features:

- All slave ports are enabled
- All slave ports have equal priority
- All slave ports have a hold time of four transactions

The following figure shows the external connections on the non-programmable ATB funnel.

Figure 5-4: `css600_atbfunnel` logical connections



5.4 ATB replicator

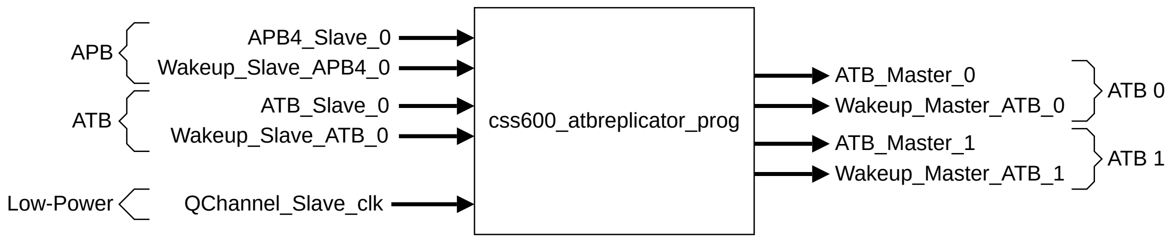
The `css600_atbreplicator` splits a single trace stream into two trace streams for systems that have more than one trace sink component.

An optional programmable configuration is available that provides the following features:

- Filtering of trace IDs to allow some IDs to go to master port 0 and some to master port 1
- Registers to allow integration testing of the trace network

The following figure shows the external connections on the programmable ATB replicator.

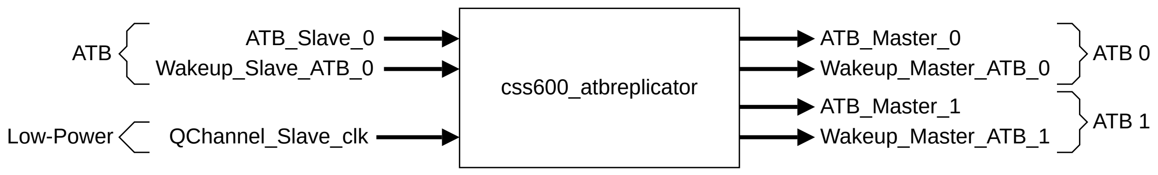
Figure 5-5: `css600_atbreplicator_prog` logical connections



In the non-programmable configuration, no ATB ID filtering is applied to either master.

The following figure shows the external connections on the non-programmable ATB replicator.

Figure 5-6: `css600_atbreplicator` logical connections



5.5 ATB trace buffer

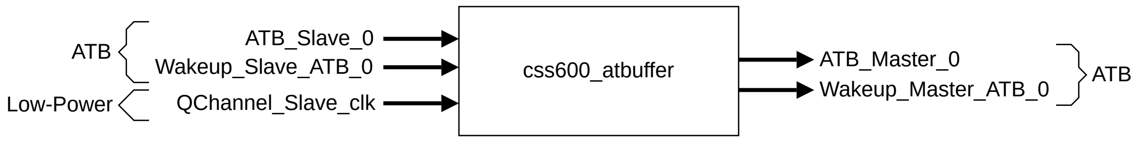
The `css600_atbbuffer` is used in situations where some local smoothing of trace bandwidth is required in a trace network.

The ATB trace buffer has the following features:

- Configurable trace data width up to 128 bits
- Configurable buffer depth up to 256 entries
- Configurable threshold for buffer fill level before starting to empty

The following figure shows the external connections on the ATB trace buffer.

Figure 5-7: `css600_atbbuffer` logical connections



5.6 ATB asynchronous bridge

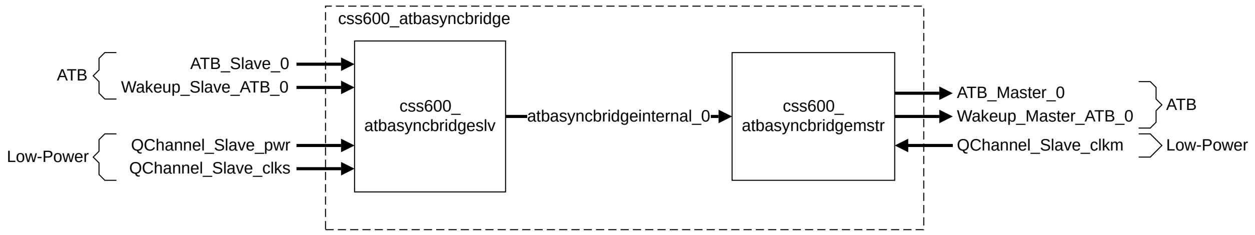
The `css600_atbasyncbridge` is used to transport the AMBA trace bus across a clock or power domain boundary.

The ATB asynchronous bridge provides the following features:

- Two independent clock domains with any phase or frequency alignment
- Two independent power domains, either of which can be switched relative to the other
- Three Q-Channel LPIs for slave side clock, master side clock, and power switching management
- Two-part meta-component with separate slave and master side components
- Configurable ATB data width
- Configurable for 2- or 3-stage synchronizers
- Automatically manages upstream flush of trace data before power down

The following figure shows the external connections on the ATB asynchronous bridge.

Figure 5-8: `css600_atbasyncbridge` logical connections



5.7 ATB synchronous bridge

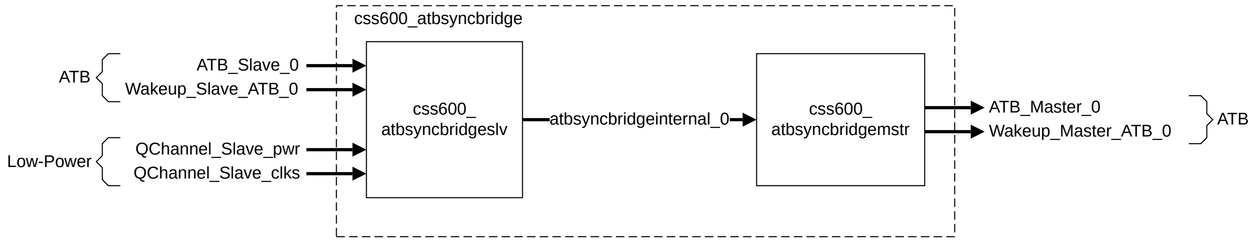
The `css600_atbsyncbridge` is used to transport the AMBA trace bus across a clock domain boundary.

The ATB synchronous bridge provides the following features:

- Two synchronous clock domains with any frequency difference. The clocks must be skew balanced and from a common source so that they are high-level-gated by a common control point.
- Two Q-Channel LPIs for clock and power switching management
- Two-part meta-component with separate slave and master side components
- Configurable ATB data width
- Configurable for 2- or 3-stage synchronizers
- Automatically manages upstream flush of trace data before power down

The following figure shows the external connections on the ATB synchronous bridge.

Figure 5-9: `css600_atbsyncbridge` logical connections



5.8 Trace Memory Controller

The css600_tmc Trace Memory Controller is used for capturing trace data into local or system memory, or streamed to an HSSTP. SoC-600M only supports configuration as an *Embedded Trace Buffer* (ETB), as this section describes.

The trace can be read by an off-chip external debugger, or by on-chip self-hosted debug software.

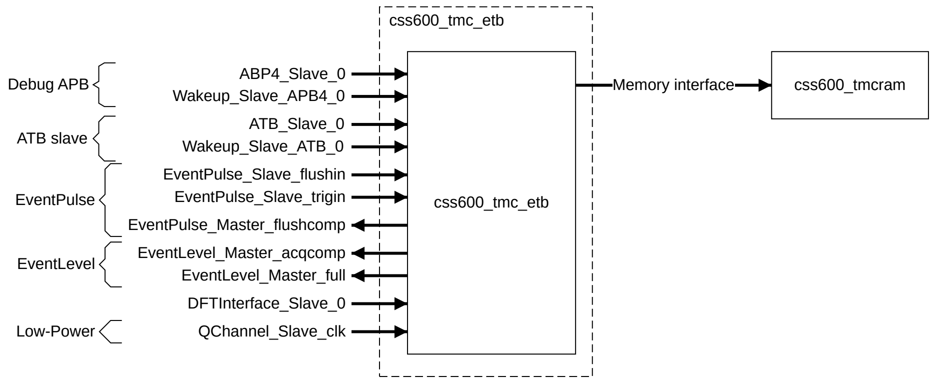
The TMC can be configured as follows:

***Embedded Trace Buffer* (ETB)**

Enables trace to be stored in a dedicated SRAM that is used as a circular buffer.

The following figure shows the external connections on the TMC ETB configuration.

Figure 5-10: `css600_tmc_etb` logical connections



The TMC can be programmed to capture trace in different modes:

Circular Buffer mode

TMC captures trace using its storage as a circular buffer, overwriting old trace when the buffer is full. In this mode, trace capture can automatically stop after receiving a trigger signal.

Software FIFO mode 1

In this mode, the component functions as a FIFO where data is read out by software over the Debug APB interface. This mode provides a low-speed communication channel for trace data, reusing the existing programming interface.

5.8.1 TMC register access dependencies

Not all TMC registers can be read and written under the same conditions.

5.8.1.1 Writes to TMC registers

You can only write to TMC registers under specific conditions.

The following table shows the conditions necessary to write to each TMC register. Writing to the TMC under conditions other than those listed results in **UNPREDICTABLE** behavior. An x indicates that any value is permitted.

Write accesses to TMC registers

Register	Name	Offset	CTL.TraceCaptEn	STS.TMCReady	MODE	ITCTRL.IME
RRD	RAM Read Data Register	`0x010`	Read-only			
RRP	RAM Read Pointer Register	`0x014`	0	1	x	0
RWP	RAM Write Pointer Register	`0x018`	0	1	x	0
TRG	Trigger Counter Register	`0x01C`	0	1	x	0
CTL	Control Register	`0x020`	x	x	x	0
RWD	RAM Write Data Register	`0x024`	0	1	x	0
MODE	Mode Register	`0x028`	0	1	x	0
LBUFLEVEL	Latched Buffer Fill Level	`0x02C`	Read-only			
CBUFLEVEL	Current Buffer Fill Level	`0x030`	Read-only			
BUFWM	Buffer Level Water Mark	`0x034`	0	1	x	0
FFSR	Formatter and Flush Status Register	`0x300`	Read-only			
FFCR.EmbedFlush	Formatter and Flush Control Register	`0x304`	x	x	x	0
FFCR.StopOnTrigEvt		`0x304`	1	x	CB	0
			0	x	x	0
FFCR.StopOnFl		`0x304`	x	x	x	0
FFCR.TrigOnFl		`0x304`	x	x	x	0
FFCR.TrigOnTrigEvt		`0x304`	1	x	CB	0
			0	x	x	0
FFCR.TrigOnTrigIn		`0x304`	x	x	x	0
FFCR.FlushMan		`0x304`	x	x	x	0
FFCR.FOnTrigEvt		`0x304`	1	x	CB	0
			0	x	x	0
FFCR.FOnFlIn		`0x304`	x	x	x	0
FFCR.EnTI		`0x304`	0	1	x	0
FFCR.EnFt		`0x304`	0	1	x	0
PSCR.PSCount	Periodic Synchronization Counter Register	`0x308`	0	1	x	0
ITEVTINTR.ACQCOMP ITEVTINTR.FULL ITEVTINTR.FLUSHCOMP	Integration Test Event & Interrupt Status Register	`0xEE0`	0	1	x	1
ITTRFLIN	Integration Test Trigger In and Flush In Register	`0xEE8`	Read-only			

Write accesses to TMC registers						
Register	**Name**	**Offset**	**CTL.TraceCaptEn**	**STS.TMCReady**	**MODE**	**ITCTRL.IME**
ITATBDATA0	Integration Test ATB Data 0 Register	`0xEEC`	Read-only			
ITATBCTR2	Integration Test ATB Control 2 Register	`0xEF0`	0	1	x	1
ITATBCTR1	Integration Test ATB Control 1 Register	`0xEF4`	Read-only			
ITATBCTR0	Integration Test ATB Control 0 Register	`0xEF8`	Read-only			
ITCTRL	Integration Mode Control Register	`0xF00`	0	1	x	x
CLAIMSET	Claim Tag Set Register	`0xFA0`	x	x	x	x
CLAIMCLR	Claim Tag Clear Register	`0xFA4`	x	x	x	x
AUTHSTATUS	Authentication Status Register	`0xFB8`	Read-only			
DEVARCH	Device Architecture Register	`0xFBC`	Read-only			
DEVID1	Device Configuration Register 1	`0xFC4`	Read-only			
DEVID	Device Configuration Register	`0xFC8`	Read-only			
DEVTYPE	Device Type Identifier Register	`0xFCC`	Read-only			
PIDR4-7	Peripheral ID Registers 4-7	`0xFD0-0xFDC`	Read-only			
PIDR0-3	Peripheral ID Registers 0-3	`0xFE0-0xFEC`	Read-only			
CIDR0-3	Component ID Registers 0-3	`0xFF0-0xFFC`	Read-only			

5.8.1.2 Reads from TMC registers

You can only read from TMC registers under specific conditions.

The following table shows the conditions under which read accesses from TMC registers return valid values. Reads at other times return **UNKNOWN** values. An x indicates that any value is permitted.

Read accesses from TMC registers							
Register	**Name**	**Offset**	**CTL.TraceCaptEn**	**STS.TMCReady**	**MODE**	**ITCTRL**	**Remarks**
RSZ	RAM Size register	`0x004`	x	x	x	x	-
STS.Empty	Status Register	`0x00C`	1	x	x	0	-
STS.FtEmpty	Status Register	`0x00C`	1	x	x	0	-
STS.TMCReady	Status Register	`0x00C`	x	x	x	0	-
STS.Triggered	Status Register	`0x00C`	1	x	CB	0	-

Read accesses from TMC registers							
Register	**Name**	**Offset**	**CTL.TraceCaptEn**	**STS.TMCReady**	**MODE**	**ITCTRL**	**Remarks**
			0	x	x	0	Value of this bit when trace capture stops is held
STS.Full	Status Register	`0x00C`	x	x	x	0	Value of this bit when trace capture stops is held
RRD	RAM Read Data Register	`0x010`	0	1	x	0	-
			1	x	SWF1	0	If trace memory is empty, the data that is returned is `0xFFFFFFFF`.
			1	1	CB	0	
RRP	RAM Read Pointer Register	`0x014`	1	x	SWF1, SWF2	0	-
			1	1	CB	0	-
			0	1	x	0	-
RWP	RAM Write Pointer Register	`0x018`	1	x	SWF1, SWF2	0	-
			1	1	CB	0	-
			0	1	x	0	-
TRG	Trigger Counter Register	`0x01C`	1	x	CB	0	The trigger counter is active only in Circular buffer mode
			0	1	x	0	
CTL	Control Register	`0x020`	x	x	x	0	-
RWD	RAM Write Data Register	`0x024`	Write-only				
MODE	Mode Register	`0x028`	1	x	x	0	-
LBUFLEVEL	Latched Buffer Fill Level	`0x02C`	1	x	x	0	-
			0	1	x	0	Value of this register when trace capture stops is held
CBUFLEVEL	Current Buffer Fill Level	`0x030`	1	x	x	0	-
			0	1	x	x	Value of this register when trace capture stops is held
BUFWM	Buffer Level Water Mark	`0x034`	x	x	x	x	Programmed registers can be read at any time. The return value is the value that was programmed.
FFSR	Formatter and Flush Status Register	`0x300`	x	x	x	0	-
FFCR	Formatter and Flush Control Register	`0x304`	x	x	x	0	-
PSCR	Periodic Synchronization Counter Register	`0x308`	x	x	x	0	-
ITEVTINTR	Integration Test Event & Interrupt Status Register	`0xEE0`	Write-only				
ITTRFLIN	Integration Test Trigger In and Flush In Register	`0xEE8`	x	x	x	1	-

Read accesses from TMC registers							
Register	Name	Offset	CTL.TraceCaptEn	STS.TMCReady	MODE	ITCTRL	Remarks
ITATBDATA0	Integration Test ATB Data Register 0	`0xEEC`	x	x	x	1	-
ITATBCTR2	Integration Test ATB Control 2 Register	`0xEF0`	Write-only				
ITATBCTR1	Integration Test ATB Control 1 Register	`0xEF4`	x	x	x	1	-
ITATBCTR0	Integration Test ATB Control 0 Register	`0xEF8`	x	x	x	1	-
ITCTRL	Integration Mode Control Register	`0xF00`	x	x	x	x	-
CLAIMSET	Claim Tag Set Register	`0xFA0`	x	x	x	x	-
CLAIMCLR	Claim Tag Clear Register	`0xFA4`	x	x	x	x	-
AUTHSTATUS	Authentication Status Register	`0xFB8`	x	x	x	x	-
DEVARCH	Device Architecture Register	`0xFBC`	x	x	x	x	-
DEVID1	Device Configuration Register	`0xFC4`	x	x	x	x	-
DEVID	Device Configuration Register	`0xFC8`	x	x	x	x	-
DEVTYPE	Device Type Identifier Register	`0xFCC`	x	x	x	x	-
PIDR4-7	Peripheral ID Registers 4-7	`0xFD0`-`0xFDC`	x	x	x	x	-
PIDR0-3	Peripheral ID Registers 0-3	`0xFE0`-`0xFEC`	x	x	x	x	-
CIDR0-3	Component ID Registers 0-3	`0xFF0`-`0xFFC`	x	x	x	x	-

5.8.2 Clock and reset

The TMC has a single clock input clk and an active-LOW reset input reset_n.

reset_n resets all interfaces and control registers except some of the memory mapped control registers. See the appropriate *Register summary* for your chosen TMC configuration for details of registers that are not initialized on reset and must be programmed before enabling TMC trace capture.

5.8.3 Interfaces

The TMC has the following interfaces:

- Debug APB interface
- ATB slave interface
- Memory interface
- Low-Power interface
- Event interfaces
- Buffer interrupt interface
- Authentication interface
- DFT interface

5.8.3.1 Debug APB interface

The Debug APB interface is used for programming the registers and to read the trace data from local SRAM.

The Debug APB interface is compliant with the AMBA APB4 protocol.

5.8.3.2 ATB slave interface

The ATB slave interface is used to receive the trace data. It can support a configurable data width, ATB_DATA_WIDTH, of 32, 64, or 128-bit.

The interface can be connected to a replicator, a trace source, or any other component with a standard ATB master. The interface complies with the *AMBA® 4 ATB Protocol Specification*.

5.8.3.3 Memory interface

The memory interface supports access to on-chip SRAM to store and retrieve trace data. Data width to memory is twice as wide as ATB width, that is 2 x `ATB_DATA_WIDTH`.

5.8.3.4 Low-Power interface

The TMC has a Q-Channel *Low-Power Interface* (LPI) for clock gating that is present in all four TMC configurations.

For more information, see the *Arm®AMBA® Low Power Interface Specification*.

5.8.3.5 Event interfaces

The TMC has five event interfaces, comprising two slave event signals, TRIGIN and FLUSHIN, and three master event signals FULL, ACQCOMP, and FLUSHCOMP, that can be connected to the *Cross Trigger Interface* (CTI).

TRIGIN

This input can cause a Trigger Event.

FLUSHIN

This input can cause a trace flush.

FULL

When the TMC is not in integration mode, this output indicates the value of the Full bit in the STS Register.

ACQCOMP

When the TMC is not in integration mode, this output indicates the value of the FtEmpty bit in the STS Register.

FLUSHCOMP

When the TMC is not in integration mode, this output pulses HIGH when a flush request is completed downstream.

You must connect these signals to a CoreSight *Cross Trigger Interface* (CTI).

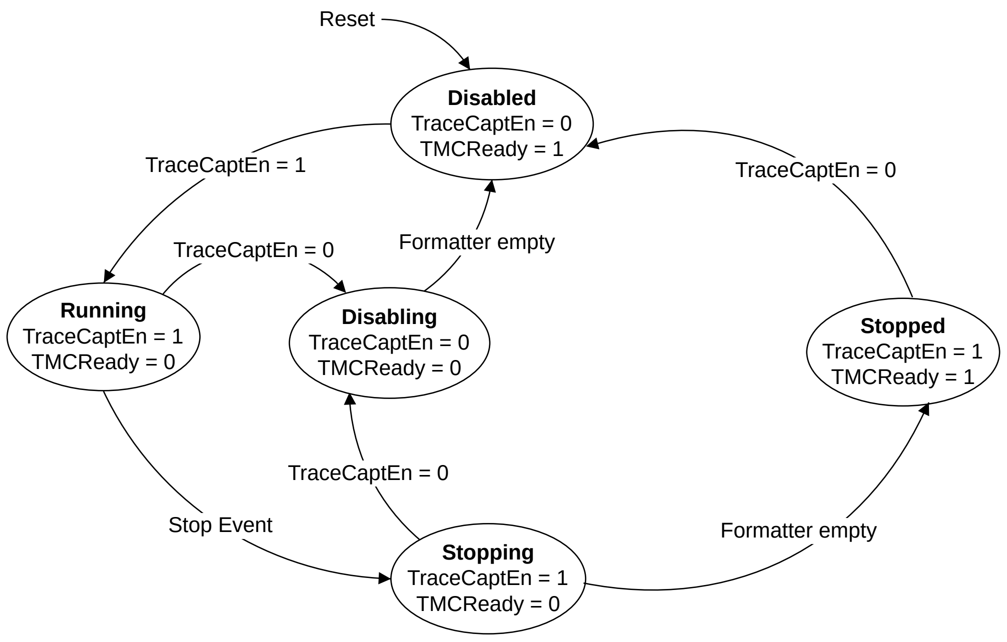
5.8.4 Operation

The TMC uses a state machine to control its operation.

5.8.4.1 Architectural state machine

The Trace Capture Enable bit, CTL.TraceCaptEn, and TMC Ready bit, STS.TMCReady, define the TMC states.

Figure 5-11: TMC architectural state machine



The operating states of Trace Memory Controller are:

5.8.4.1.1 DISABLED: (CTL.TraceCaptEn=0, STS.TMCReady=1)

DISABLED is the default state of TMC after reset and whenever CTL.TraceCaptEn is cleared. All programming must be performed in this state.

When the TMC is in DISABLED state, the contents of most registers, including the MODE and FFCR registers, have no effect. For backwards compatibility, the contents of the circular buffer can be read in this state. The debugger must manually manage the read pointer. The TMC enters RUNNING state from DISABLED state when the CTL.TraceCaptEn bit is set.

5.8.4.1.2 RUNNING: (CTL.TraceCaptEn=1, STS.TMCReady=0)

RUNNING is the functional state during which trace capture is performed.

The STOPPING state is entered from this state when a Stop event occurs.

5.8.4.1.3 STOPPING: (CTL.TraceCaptEn=1, STS.TMCReady=0)

In STOPPING state, the TMC begins to drain the trace data from its internal pipelines to the trace memory.

From the programmers model, the STOPPING state is indistinguishable from the RUNNING state. The STOPPED state is entered from STOPPING state when the following conditions are true:

- All trace has been output, including null padding, if necessary, to drain the last few bytes of trace, and the formatter and write buffer are empty.
- In SWF1 mode, there must be space in the FIFO for data that is left in the pipeline to be written to the FIFO. To achieve this, it might be necessary to read extra data from the FIFO. If no space is available in the FIFO, then the STOPPED state is not reached.

5.8.4.1.4 STOPPED: (CTL.TraceCaptEn=1, STS.TMCReady=1)

In STOPPED state, no trace capture takes place, but data that is still in the trace memory can be read out.

When in STOPPED state:

- In CB mode, except when streaming, the captured trace can be read out over debug APB.
- In SWF 1, the remaining contents of the FIFO can be read out over debug APB.

The DISABLED state is entered from this state by clearing CTL.TraceCaptEn bit.

5.8.4.1.5 DISABLING: (CTL.TraceCaptEn=0, STS.TMCReady=0)

DISABLING is an *emergency stop* state that can be entered at any time by clearing CTL.TraceCaptEn.

DISABLING state differs from the STOPPING state in the following ways:

- The next transition is to the DISABLED state, not the STOPPED state. This transition means that:
    - Unretrieved trace is lost.
- Exit from DISABLING state is not dependent on reads performed from the RRD register in SWF1 mode.

Arm recommends that the trace capture is stopped by programming an appropriate STOP event in the FFCR register. For example, trace capture can be stopped by setting the FFCR.StopOnFl bit, and then initiating a manual flush by setting the FFCR.FlushMan bit.

The *emergency stop* option is provided so that the TMC is programmer-compatible with the ETB that was delivered with SoC-400M, where the only way to stop trace capture was by clearing CTL.TraceCaptEn bit. Use of the emergency stop is otherwise discouraged, especially in SWF1 mode, where it can lead to loss of trace or even trace corruption.

5.8.4.2 Formatter and stop sequence

When EnFt in the FFCR is set, formatting is enabled.

For more information about the formatting protocol, see the *CoreSight Architecture Specification*.

Depending on the configuration of the TMC, trace might be written up to 256 bits at a time. Additionally, when the formatter is enabled by setting the EnFt bit in the FFCR, a whole number of frames must be written. When stopping trace capture, the TMC pads the end of the trace so that every byte of trace that has been accepted by the TMC is written.

Note

The stop sequence might be longer than required to meet the rules, to simplify the implementation.

5.8.4.2.1 Formatter enabled

If the formatter is enabled when trace capture is stopped, the traces are padded in the formatted frames with extra bytes of data with a value of `0x00` and an ID of `0x00`.

The traces are padded until the following conditions are met:

- A whole number of frames have been generated.
- The trace is aligned to the memory width. This means that if the ATB interface is configured to 128 bits, then a multiple of two frames has been generated to meet the 256-bit memory width.

5.8.4.2.2 Formatter disabled

Disabling the formatter is deprecated, and is supported in CB mode only.

If the formatter is disabled when trace capture is stopped, then the trace is padded with additional bytes so that the precise end of the trace can be determined, as follows:

- A single byte of value `0x01`, to indicate the position of the last byte before the stop sequence
- Zero or more bytes of value `0x00`, to align to the memory width

5.8.4.3 Trigger, flush, and stop events

The *Formatter and Flush Control Register* (FFCR) includes controls for the following:

- Enabling the formatter to wrap data from multiple trace sources into frames CoreSight Architecture Specification v1.0 describes
- The insertion of trigger markers into the formatted trace stream

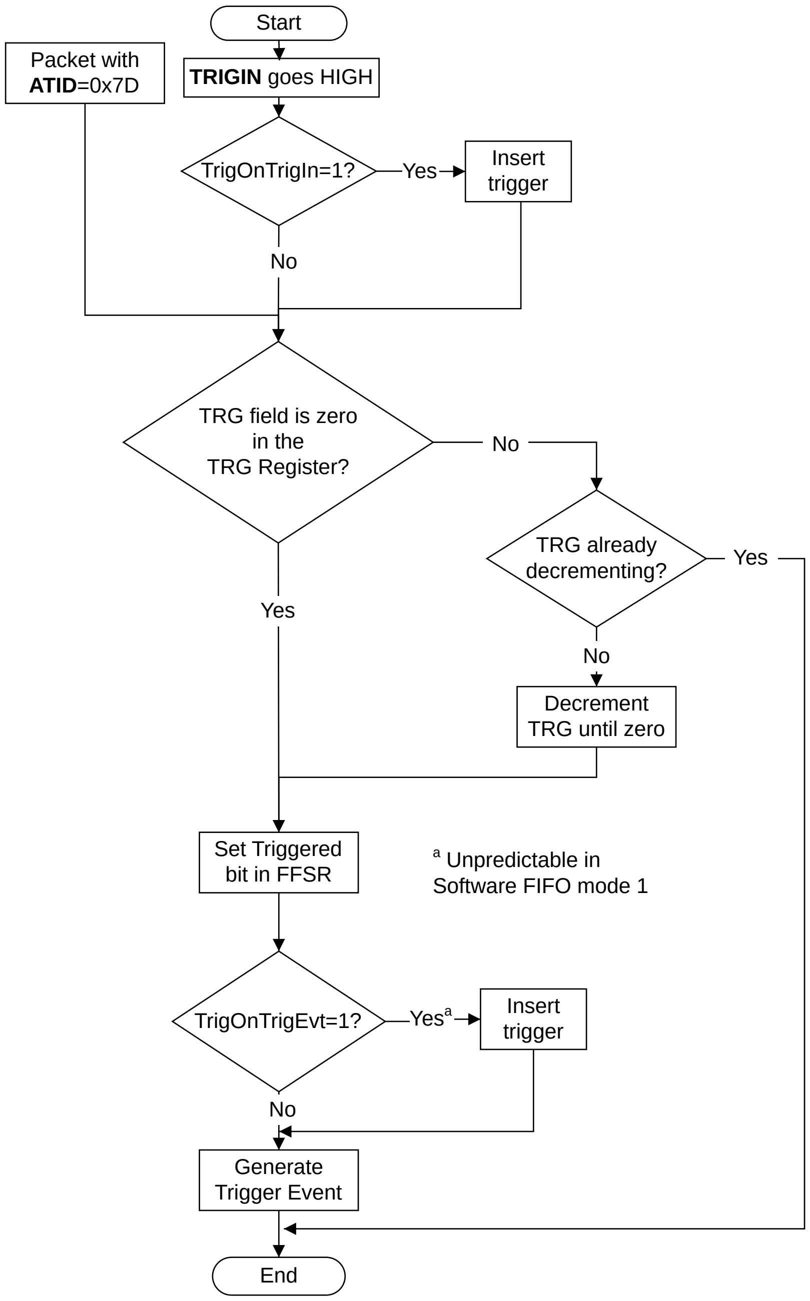
- When to stop trace capture
- When to perform a flush

5.8.4.3.1 TRIGIN and ATB slave interface trigger

The following figure shows the actions taken on an input trigger.

The input trigger is either when an event is sampled on TRIGIN, or when a packet is accepted on the ATB slave interface when ATID is equal to`0x7D`.

Figure 5-12: Actions taken on input trigger



5.8.4.3.2 Flush and stop

A Trigger Event can cause a flush, can stop the TMC, and can cause another trigger insertion.

The following table shows the TMC response to these events.

Table 5-3: Event generation

Event	MODE	Outcome
FlushMan or (FOnFlIn & FLUSHIN)	CB	Flush the trace sources that feed the TMC: • Insert a trigger if TrigOnFl is set in the FFCR. • Stop trace capture if StopOnFl is set in the FFCR.
	SWF1	Flush the trace sources that feed the TMC, and ensure that the flushed trace is ready to be read by subsequent reads of the RRD register: • Insert a trigger if TrigOnFl is set in the FFCR. • Stop trace capture if StopOnFl is set in the FFCR.
Trigger Event	CB	If StopOnTrigEvt is set in the FFCR, then stop trace capture. If StopOnTrigEvt is not set in the FFCR: • Insert a trigger if TrigOnFl is set in the FFCR. • Stop trace capture if StopOnFl is set in the FFCR.
	SWF1	Ignored
TraceCaptEn bit cleared during trace capture	Any	Stop trace capture immediately. The captured trace is lost.

Note

The TMC always outputs any outstanding triggers in the trace before completing a flush or stopping trace capture. A rapid stream of triggers on TRIGIN or on the ATB slave interface can cause the flush or stop to be delayed.

5.8.4.3.3 Common usage

Many bits can be set simultaneously, leading to a wide range of programming settings, not all of which are useful.

In practice, the most common setting in CB mode is to set the TrigOnTrigIn, FOnTrigEvt, StopOnFl, EnTI and EnFt bits in the FFCR.

1. Wait for a trigger.
2. Insert a trigger into the trace stream.
3. Count down the trigger counter.
4. Flush.
5. Stop trace capture.

5.8.5 Standard usage models

This section describes the recommended usage model for the TMC in several operating modes.

5.8.5.1 CB mode

The recommended standard usage model, with optional manual stop, is as follows:

1. Wait until TMCReady in the STS register is equal to one.
2. Program the MODE Register for CB mode.
3. Program the FFCR Register. Arm recommends that you set the TrigOnTrigIn, FOnTrigEvt, StopOnFl, EnTI, and EnFt bits. This enables formatting, inserting a trigger when a trigger is observed on TRIGIN, and following a delay corresponding to the value of the TRG Register, flushing and then stopping the TMC.
4. Program the TRG Register, to control the amount of buffer to be dedicated to the period after a trigger is observed.
5. Write RWP and write RRP with the same value as RWP. Arm recommends that RWP=0.
6. Set the TraceCaptEn bit in the CTL Register. This starts the trace session.
7. Either:
    - Wait until TMCReady in STS register is equal to one. This indicates that the trace session is over, because a trigger event has occurred.
    - To stop capture manually without a trigger event, set the FlushMan bit in the FFCR REgister and then wait until TMCReady is equal to one. This indicates that the trace session is over because the manual flush completed.
8. Read the contents of the trace buffer by performing successive reads to the RRD Register, until the value `0xFFFFFFFF` is returned.
9. Clear the TraceCaptEn bit in the CTL Register.

5.8.5.2 SWF1 mode

This recommended standard usage mode is as follows:

1. Wait until TMCReady in the STS register is equal to one.
2. Program the MODE Register for SWF1 mode.
3. Program the FFCR Register. Arm recommends that you set the TrigOnTrigIn, FOnTrigEvt, StopOnFl, EnTI, and EnFt bits. This enables formatting, inserting a trigger when a trigger is observed on TRIGIN, and following a delay corresponding to the value of the TRG Register, flushing and then stopping the TMC.
4. Program the BUFWM Register to the required threshold fill level. You can usually set this register to zero.
5. Program the TRG Register to control the amount of buffer to be dedicated to the period after a trigger is observed.
6. Write RWP and write RRP with the same value as RWP. Arm recommends that RWP=0.
7. Set the TraceCaptEn bit in the CTL Register. This starts the trace session.

8. Start reading data from the RRD Register. If the value `0xFFFFFFFF` is returned, then no data is available, and the read must be retried. Continue until the trace session is over, for example, following receipt of a trigger in the trace stream.
9. Either:
    - Read data from the FIFO to get flushed data, retrying when `0xFFFFFFFF` is returned, until TMCReady is equal to one. This indicates that the trace session is over, because a trigger event has occurred.
    - To stop capture manually without a trigger event, set the FlushMan bit in the FFCR Register to flush, then wait until TMCReady is equal to one. This indicates that the trace session is over because the manual flush completed.

```
Repeat {
  Read Data from RRD register
  If (Data = 0xFFFFFFFF) {
 Read TMCReady from STS register
 If (TMCReady = 1) {
    Stop
 }
  } else {
 Add Data to the end of the trace
  }
}
```

10. Read the remaining data from the FIFO, stopping when `0xFFFFFFFF` is returned. This indicates that the FIFO is empty.
11. Clear the TraceCaptEn bit in the CTL Register.

5.9 Trace Port Interface Unit

An external *Trace Port Analyzer* (TPA) captures trace data when the TPIU drives the external pins of a trace port.

The TPIU does the following:

- Coordinates stopping trace capture when a trigger is received
- Inserts source ID information into the trace stream so that trace data can be re-associated with its trace source. The operation of the trace formatter is described in the *CoreSight™ Architecture Specification*
- Outputs the trace data over trace port pins
- Outputs patterns over the trace port so that a TPA can tune its capture logic to the trace port, maximizing the speed at which trace can be captured

5.9.1 Clocks and resets

The clock and reset signals of the TPIU are clk, tranclk_in, reset_n, and treset_n.

The TPIU includes an asynchronous bridge between the traceclk_in clock domain and the rest of the design.

An external APB asynchronous bridge can be used to bridge to another clock domain if necessary.

5.9.2 Functional interfaces

This section describes the functional interfaces of the TPIU.

The functional interfaces are:

ATB slave interface
: Receives trace data

APB slave interface
: Accesses the TPIU registers

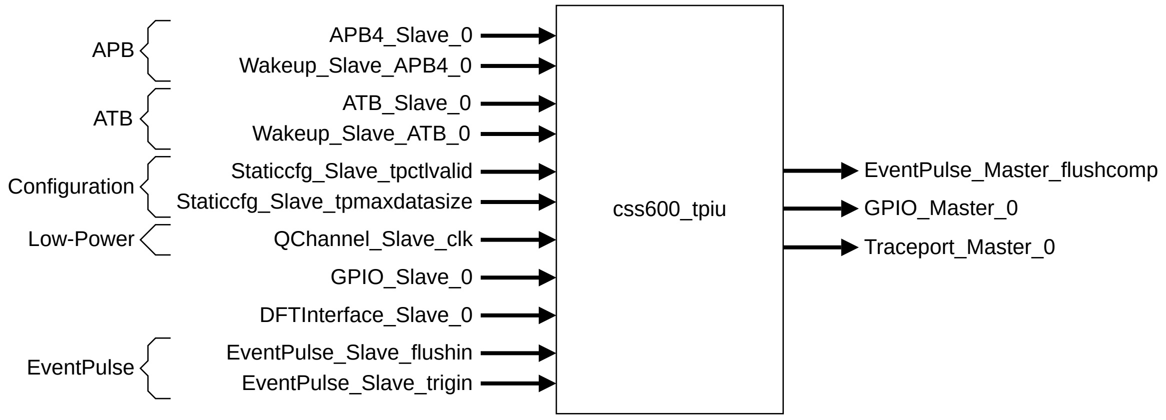
Trace out port
: Connects to the external trace port pins

trigin and flushin event interfaces
: Implement synchronizers so that they can be connected to a CTI in a different clock domain

The TPIU also supports the extctlin[7:0] and extctlout[7:0] signals. These signals are designed to enable debug tools to multiplex the pins used by the trace out port with other functions.

The following figure shows the external connections of the TPIU.

Figure 5-13: Trace Port Interface Unit block diagram



5.9.3 Parallel trace out port

This section describes the parallel trace out port signals.

To select parallel trace mode you must program TPIU_SPPR. TPIU_SSPSR indicates the supported parallel trace port widths. To select the active parallel trace port width, program TPIU_CSPSR. You can vary the data rate of the interface by programming TPIU_ACPR.

5.9.3.1 Signals of the trace out port

The following table summarizes the trace out port signals.

Table 5-4: Trace out port signals

Name	Type	Description
traceclk	Output	Output clock, that the TPA uses to sample the other pins of the trace out port. This signal runs at half the speed of traceclk_in, and data is valid on both edges of this clock.
tracedata[31:0]	Output	Output data. A system might not connect all the bits of this signal to the trace port pins. The connection depends on the number of pins available and the bandwidth that is required to output trace.
tracectl	Output	Signal to support legacy TPAs which cannot support formatter operation in continuous mode. Connection of this signal to a pin is optional.

For information on the configuration tie-off signals, see the *Arm® CoreSight™ SoC-600M Configuration and Integration Manual*.

5.9.4 traceclk alignment

The TPIU does not offset the edges of traceclk from the edges of the trace data signals tracedata and tracectl. For compatibility with the maximum number of TPAs, Arm recommends that you delay tracectl so its edges are in the middle of the stable phases of the data signals.

Arm recommends that TPAs support systems with a variety of alignments of traceclk relative to the data signals. This recommendation includes systems where edges of traceclk occur at the same time as transitions of the data signals. Having a variety of alignments supports the widest range of targets at the maximum speed.

5.9.5 tracectl removal

The TPIU supports traceclk + tracedata + tracectl, with a minimum tracedata width of 2, and traceclk + tracedata, with a minimum data width of 1.

The chosen mode depends on the connected trace port analyzer or capture device. Legacy capture devices use the control pin to indicate when there is valid data to capture. Newer capture devices can use more pins for data and do not require a reserved tracectl pin.

If support for legacy TPAs is not required, it is not necessary to implement the tracectl pin. This design choice must be reflected by the value of tpctl.

5.9.6 tracectl encoding

You can implement tracectl so that bypass mode or normal mode is selected in the Formatter and Flush Control Register. In this case, the tracectl and tracedata[1:0] encodings are designed to be backwards compatible with systems designed without CoreSight™, where a trace port is driven by a single ETM.

The encodings indicate to the TPA:

- Whether the trigger has occurred. This can be used by the TPA to stop trace capture, when the TPA is responsible for stopping trace capture.
- Whether to capture data from the trace port in this cycle.

5.9.7 Trace port triggers

The TPIU trace port is backwards compatible with non-CoreSight™ systems where a single ETM drives the trace port. Compatibility is achieved when tracectl is implemented and bypass or normal mode is selected in the Formatter and Flush Control Register, FFCR.

The trigger is an indication to the TPA to stop trace capture. In CoreSight™ systems, the TPIU receives trigger events from trace sources through the cross-triggering system. The TPIU sends a trigger event over the trace out port to the TPA when it is ready for trace capture to stop.

The TPIU might signal a trigger as a result. This trigger can be:

- Directly from an event such as a pin toggle from the CTI
- A delayed event such as a pin toggle that has been delayed coming through the Trigger Counter Register
- The completion of a flush

The following table extends the *Embedded Trace Macrocell™ ETMv1.0 to ETMv3.5 Architecture Specification* on how a trigger is represented.

Table 5-5: CoreSight™ representation of triggers

tracectl	tracedata		Trigger	Capture	Description
	[1]	[0]	Yes/No	Yes/No	
0	x	x	No	Yes	Normal trace data
1	0	0	Yes	Yes	Trigger packet. The trigger packet encoding is not generated by the TPIU.
1	1	0	Yes	No	Trigger
1	x	1	No	No	Trace disable

5.9.7.1 Correlation with afvalid

When the TPIU receives a trigger signal, it can request a flush of all trace components through the ATB slave interface.

This flush request depends on the settings in the Formatter and Flush Control Register. The flush request causes all information around the trigger event to be flushed from the system before normal trace information is resumed. The flush ensures that all information that is related to the trigger is output before the TPA, or other capture device, is stopped.

With FOnTrig set to 1, it is possible to indicate the trigger on completion of the flush routine. Knowing the trigger ensures that if the TPA stops the capture on a trigger, the TPA gets all historical data relating to the trigger.

5.9.8 Programming the TPIU for trace capture

The following points must be considered when programming the TPIU registers for trace capture.

- TPAs that are only capable of operation with tracectl must only use the formatter in either bypass or normal mode, not in continuous mode.
- Arm recommends that, following a trigger event within a multi-trace source configuration, a flush is performed to ensure that all historical information that is related to the trigger is output.
- If Flush on Trigger Event and Stop on Trigger Event options are chosen, then the TPA does not capture any data after the trigger. When the TPIU is instructed to stop, it discards any subsequent trace data, including data returned by the flush. Select Stop on Flush completion instead.
- Multiple flushes can be scheduled using Flush on Trigger Event, Flush on flushin, and manual flush. When one of these requests is made, it masks more requests of the same type. This masking means that repeated writing to the manual flush bit does not schedule multiple manual requests unless each is permitted to complete first.
- Unless multiple triggers are required, it is not advisable to set both Trigger on Trigger Event and Trigger on Flush Completion, if Flush on Trigger Event is also enabled. In addition, if Trigger on trigin is enabled with this configuration, it can also cause multiple trigger markers from one trigger request.
- Arm recommends that you only enable the pattern generator while the formatter is stopped.

5.9.9 Example configuration scenarios

This section contains example configuration scenarios.

The example scenarios are:

- Capturing trace after an event, and stopping
- Only indicating triggers, and continuing to flush
- Multiple trigger indications
- Independent triggering and flushing

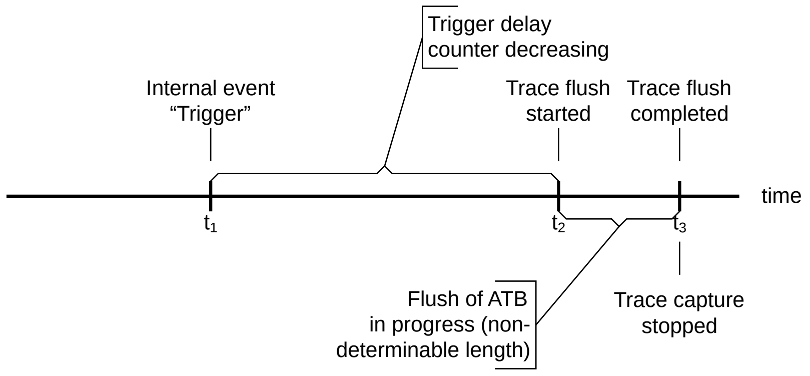
5.9.9.1 Capturing trace after an event, and stopping

A minimum amount of time must elapse before a trace capture can be stopped.

The elapsed time between the trigger and the stopping of the trace must be long enough to allow the trace data to progress through the system. Any historical information, relating to previous events, must have been emitted.

The following figure shows a possible time-line of events where an event of interest, referred to as a trigger event, causes some trace to be captured. When the trace has been captured, the trace capture device can be stopped.

Figure 5-14: Capturing trace after an event and stopping



When one trace source is used, you do not have to flush the system. Instead, you can increase the length of the trigger counter delay to enable more trace to be generated, effectively pushing out historical information.

The trigger event at time t1 is signaled to the TPIU through the cross-triggering system. The trace source that generated the trigger event might also embed some trigger information in its trace stream at this point.

The TPA only registers a trigger at time t3, when it is safe to stop trace capture. If the TPIU is in bypass or normal mode, it embeds a trigger in the formatted trace stream at time t3, and signals a trigger on tracectl.

In the figure, the action that causes trace capture to be stopped at time t3 can be one of the following:

- The TPA can watch for a trigger to be indicated through tracectl and stop.
- The TPA can watch for a trigger to be indicated in the tracedata stream, using continuous mode without the requirement for tracectl.
- The TPIU can automatically stop trace after it has signaled the trigger to the TPA.

5.9.9.2 Only indicating triggers, and continuing to flush

You can indicate a trigger at the soonest possible moment, and cause a flush, while at the same time permitting externally requested flushes.

This ability enables trace around a key event to be captured, and all historical information to be stored within a period immediately following the trigger. Use a secondary event to cause regular trace flushes.

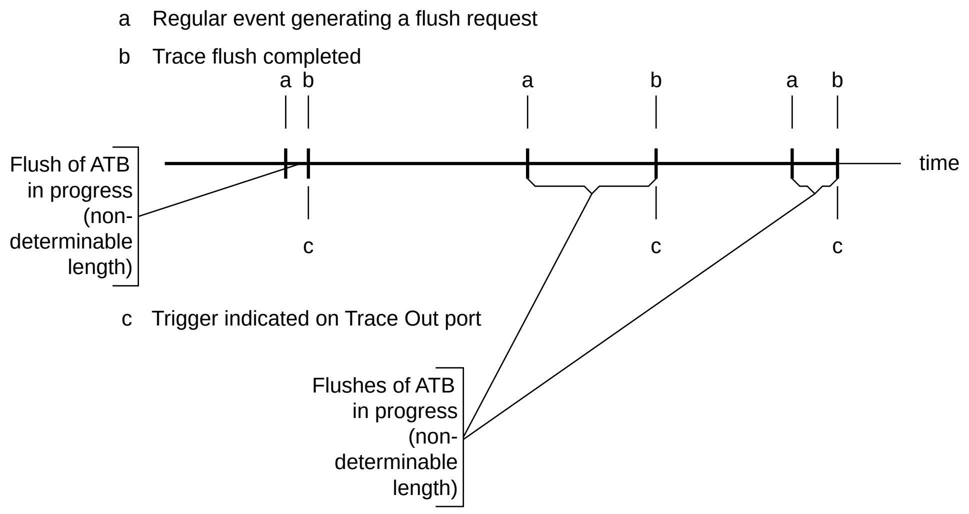
5.9.9.3 Multiple trigger indications

Sending a trigger to external tools can have extra consequences apart from stopping trace capture.

For example, this can be in cases where the events immediately before the trigger might be important, but only a small buffer is available. In this case, uploads to a host computer for decompression can occur, reducing the amount of trace data that is stored in the TPA. This procedure is also useful where the trigger originated from a device that is not directly associated with a trace source, and is a marker for a repeating interesting event.

The following figure shows multiple trigger indications from flushes.

Figure 5-15: Multiple trigger indications from flushes


a Regular event generating a flush request
b Trace flush completed
a b
a
b
a b
time
Flush of ATB in progress (non-determinable length)
c
c
c
c Trigger indicated on Trace Out port
Flushes of ATB in progress (non-determinable length)

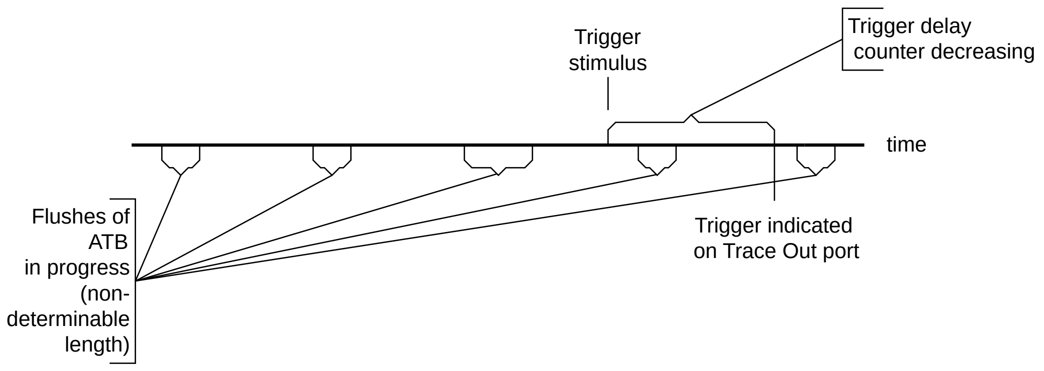
5.9.9.4 Independent triggering and flushing

The TPIU has separate inputs for flushes and triggers.

Although flushes can generate triggers, and triggers generate flushes, there might be a requirement to keep them separate. A timing block that is connected to a CTI can provide a consistent flow of new information through the Trace Out port by scheduling a regular flush. These regular events must not be marked in the trace stream as triggers.

Special events coming through the CTI that require a marker must be passed through the trigin pin. These events can either be indicated immediately or, as the following figure shows, they can be delayed through other flushes and then indicated to the TPA.

Figure 5-16: Independent triggering during repeated flushes


Trigger stimulus
Trigger delay counter decreasing
time
Flushes of ATB in progress (non-determinable length)
Trigger indicated on Trace Out port

5.9.10 TPIU pattern generator

A simple set of defined bit sequences or patterns can be output over the trace port. The TPA, or other associated trace capture device, can detect these sequences.

Analysis of the output can indicate whether it was possible to increase or, for reliability, to decrease the trace port clock speed. To ensure reliable data capture, the patterns can be used to determine the timing characteristics. The patterns can also be used for measuring the timing characteristics on the data channels to the TPA to determine whether data can be captured reliably.

Note

Arm recommends that you only enable the pattern generator while the formatter is stopped.

5.9.10.1 Pattern generator modes of operation

To enable various metrics to be determined, several patterns are supported.

The metrics can include:

- Timing between pins
- Data edge timing
- Voltage fluctuations
- Ground bounce
- Cross talk

When examining the trace port, you can choose from the following pattern modes:

Timed Each pattern runs for a programmable number of traceclk_in cycles. After completing the programmed number of cycles, the pattern generator unit

	reverts to an off state where normal trace is output, assuming trace output is enabled. The first thing that the trace port outputs after returning to normal trace is a synchronization packet. This behavior is useful with special trace port analyzers and capture devices that are aware of the pattern generator. The TPIU can be set to a standard configuration that the capture device expects. The preset test pattern can then be run, after which the TPA is calibrated ready for normal operation. The TPIU switches to normal operation automatically, without the requirement to reprogram the TPIU.
Continuous	The selected pattern runs continuously until manually disabled. This behavior is primarily intended for manual refinement of electrical characteristics and timing.
Off	When neither of the other two modes is selected, the device reverts to outputting any trace data. After timed operation finishes, the pattern generator reverts to the Off mode.

5.9.10.2 Supported options

Patterns operate over all the tracedata pins for a given port width setting.

Test patterns are aware of port sizes and always align to tracedata[0]. Walking bit patterns wrap at the highest data pin for the selected port width even if the device has a larger port width available. Also, the alternating patterns do not affect disabled data pins on smaller trace port sizes.

5.9.10.2.1 Walking 1s

All output pins are clear, with only a single bit set at a time, tracking across every tracedata output pin.

You can use this pattern to:

- Watch for data edge timing, synchronization, high-voltage level of logic 1, and cross talk against adjacent wires
- Test for broken or faulty cables and data signals

5.9.10.2.2 Walking 0s

All output pins are set, with only a single bit cleared at a time, tracking across every tracedata output pin.

Like the walking 1s, you can use walking 0s to watch for data edge timing, synchronization, low voltage level of logic 0, cross talk, and ground lift.

5.9.10.2.3 Alternating 55/AA pattern

Alternate tracedata pins are set with the others clear. This alternates every cycle.

The pattern is:

1. tracedata[0] set to 55 pattern = `0b0101_0101`
2. tracedata[1] set to AA pattern = `0b1010_1010`

This pattern repeats over the entire selected bus width. You can use this pattern to check voltage levels, cross talk, and data edge timing.

5.9.10.2.4 Alternating FF/00 pattern

On each clock cycle, the tracedata pins are either all set FF pattern or all cleared `0x00` pattern. The drivers are under various stresses. Alternating the whole set of data pins like this is a good way to check the power supply stability to the TPIU and the final pads.

6. Timestamp components functional description

This chapter describes the functionality of the timestamp components.

6.1 Timestamp generator

The `css600_tsgen` timestamp generator is used to generate a 64-bit rolling time for distribution to other CoreSight™ components that are used to align trace information.

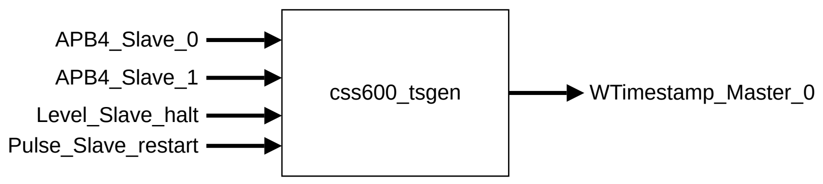
The timestamp generator has two APB interfaces: a read-only interface for reading the counter value and management registers, and a programming interface, which is designed to be accessible only to Secure software.

The counter in the timestamp generator also has the following key features:

- It runs at a constant clock frequency, regardless of the power and clocking state of the processor cores that are using it.
- When enabled and running, it can increment by 1 only.
- The counter continues to run in all levels of power down, other than when the system is turned off.
- The counter starts from 0.
- The counter value can be read using a 32-bit read on an APB interface.
- The counter value can be written only when it is either halted or disabled.
- When the system is halted as a result of debug, the counter can be programmed to either halt or continue incrementing.

The following figure shows the external connections on the Timestamp generator.

Figure 6-1: `css600_tsgen` logical connections



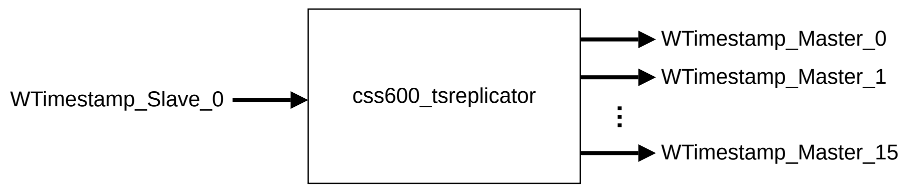
6.2 Timestamp replicator

The `css600_tsreplicator` is an IP-XACT phantom component that is provided to support stitching in an IP-XACT tooling product. There is no Verilog module for `css600_tsreplicator`.

Use the `css600_tsreplicator` to connect a single *Wide Timestamp* (WTS) master interface to multiple WTS slave interfaces. This is useful when you distribute WTS to multiple slaves in the same clock domain without the additional logic cost of a *Narrow Timestamp* (NTS) solution, and where the wire count of WTS is acceptable.

The following figure shows the external connections on the Timestamp replicator.

Figure 6-2: `css600_tsreplicator` logical connections



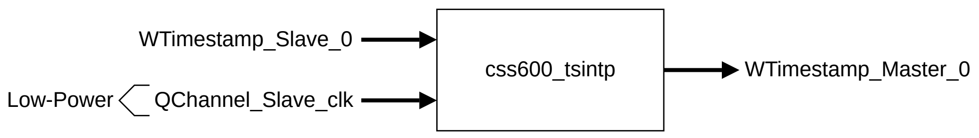
6.3 Timestamp interpolator

The timestamp interpolator increases the resolution of a timestamp.

The interpolator shifts the input timestamp left by 8 bits, and uses the extra low-order bits to provide a more accurate timestamp value. The greater accuracy is achieved by monitoring changes to the input timestamp value over time to predict how fast it counts.

The following figure shows the external connections on the Timestamp interpolator.

Figure 6-3: `css600_tsintp` logical connections



6.3.1 Functional interface

The timestamp interpolator adjusts to changes in the rate of the incoming timestamp.

The interpolator ensures that the interpolated timestamp never counts backwards, and pauses incrementing the interpolated timestamp if it gets ahead of the input timestamp value.

6.3.2 Low-Power Interface

The timestamp interpolator has a *Low-Power Interface* (LPI) to manage power reduction using high-level clock gating.

If the clock to the interpolator must be gated off, then the clock controller must use the LPI. When the interpolator exits the low-power state, it automatically recalculates the interpolation ratio before advancing the interpolated timestamp.

6.3.3 Limitations

Use of the timestamp interpolator is subject to some limitations.

The limitations are:

- The timestamp interpolator must not be used in the timestamp network that is used to distribute processor time.
- There must be only one timestamp interpolator between the timestamp generator and a component that receives the timestamp.

7. Embedded Cross Trigger components functional description

This chapter describes the functionality of the *Embedded Cross Trigger* (ECT) components.

7.1 About cross triggering

The cross-triggering components enable CoreSight™ components to broadcast events between each other.

Events are distributed as follows:

- Each event type is connected to a trigger input on a *Cross Trigger Interface* (CTI).
- Each CTI can be programmed to connect each trigger input to each of four channels. If programmed to do so, it causes an event on the corresponding channel when an input event occurs.

CTIs are connected to each other using one or more *Cross Trigger Matrices* (CTMs), through channel interfaces. When an event occurs on a channel, it is broadcast on that channel to all other CTIs in the system.

Each CTI can be programmed to connect each channel to each of several trigger outputs. If programmed to do so, it causes an event on the trigger output when a channel event occurs.

Each CTI trigger output can be connected to a CoreSight™ component event input.

Cross triggering can take place between trigger inputs and outputs on a single CTI, or between multiple CTIs. CTIs can be programmed not to broadcast events for selected channels, so that certain events can only trigger output events on the same CTI. Only the CTIs are programmable, not the CTMs.

7.2 Event signaling protocol

The cross-triggering system does not attempt to interpret the events that are signaled through it.

Events between CTI components and debug system components are transmitted using one of three mechanisms.

For CTI input events:

1. The event is signaled as a single-cycle clock pulse - an EventPulse. This option is selected using the CTI event configuration option `EVENT_IN_LEVEL` = 0.
2. The event is signaled from the debug components as a request-acknowledge signal pair and signaled to the CTI input as a single-cycle EventPulse. This option is selected using the CTI event configuration option `EVENT_IN_LEVEL` = 0. Connecting the debug component event

request-acknowledge signals to the CTI event input using the master side of an Event Pulse asynchronous bridge.

3. The event is signaled from the debug component as a level sensitive event - an EventLevel. This option is selected using the CTI event configuration option `EVENT_IN_LEVEL` = 1.

For CTI output events:

1. The event is signaled as a single-cycle clock pulse - an EventPulse. This option is selected using the CTI event configuration option `SW_HANDSHAKE` = 0.
2. The event is signaled to the debug components as a request-acknowledge signal pair and signaled from the CTI output as a single-cycle EventPulse. This option is selected using the CTI event configuration option `SW_HANDSHAKE` = 0 and connecting the debug component event request-acknowledge signals to the CTI event output using the slave side of an Event Pulse asynchronous bridge.
3. The event is signaled to the debug component as a level sensitive event - an EventLevel. This option is selected using the CTI event configuration option `SW_HANDSHAKE` = 1.

Events are broadcast between CTI and CTM components on the cross-trigger channels as a pulse. When an event passes across a clock domain boundary using an asynchronous bridge, handshaking occurs to ensure that the event lasts for exactly one clock cycle in the destination clock domain.

Each channel is a shared broadcast medium that can carry events from multiple sources going to multiple domains. When a CTI sends events onto a channel, they can coincide with other events on the same channel, so that the events become pulses of more than one clock cycle. This behavior is normal within the cross trigger system.

In usage models that count events that are passed through the cross-triggering system, events that occur close together might be merged into a single event with a single pulse when passed to another clock domain.

7.3 Cross Trigger Interface

The `css600_cti` connects one or more event sources and one or more event destinations to the cross trigger network.

The CTI has the following functional interfaces:

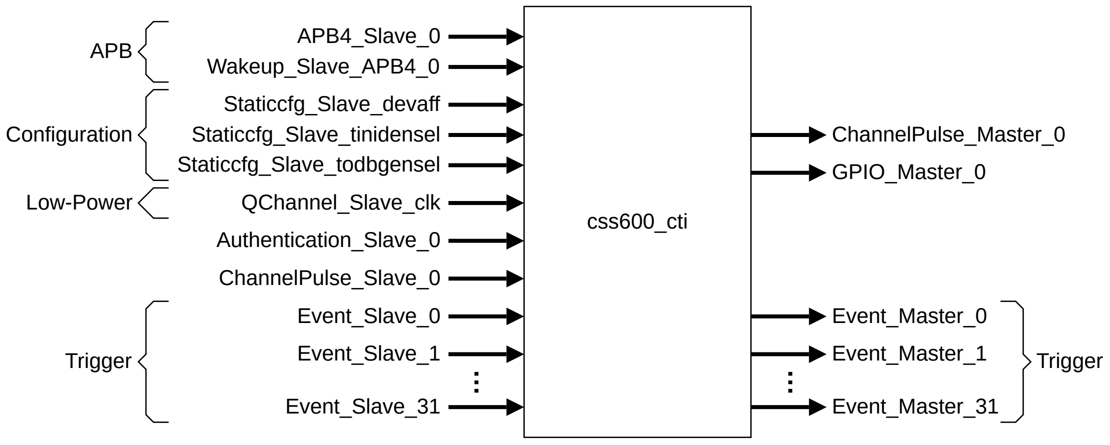
- Up to 32 trigger inputs, enabling events to be signaled to the CTI
- Up to 32 trigger outputs, enabling the CTI to signal events to other components
- A channel interface for connecting CTIs together using one or more CTMs
- An APB interface for accessing the registers of the CTI
- An Authentication interface for controlling access to certain debug events
- Eight asicctrl signals that can be used to control external multiplexers

The CTI includes configuration tie-off inputs that enable several trigger input and output types to be connected.

Arm recommends that the CTI that is connected to a processor is disabled before the processor clock is stopped. This operation minimizes the likelihood of unexpected events entering the cross-triggering system or affecting the processor when its clock is restarted. The CTICONTROL.CTIEN register bit can be used to disable the CTI globally without changing the event mapping programming.

The following figure shows the external connections on the Cross Trigger Interface.

Figure 7-1: `css600_cti` logical connections



7.3.1 asicctrl

The asicctrl output of the CTI can be used to control multiplexing on a CTI event input if necessary.

The exact configuration of any external multiplexing is user-defined. Arm does not define any relationship between values that are written to the control register and the actual configuration of the multiplexers.

The system integrator sets the `EXT_MUX_NUM` parameter to indicate the configuration of any external multiplexers. Arm does not specify the usage of the `EXT_MUX_NUM` parameter. See your system integrator for details of the implementation of your specific SoC.

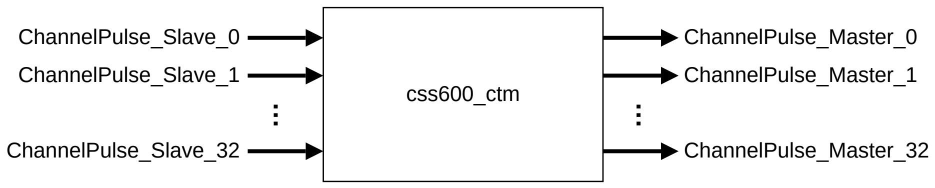
7.4 Cross Trigger Matrix

The `css600_ctm` is used to connect CTI components together in a cross trigger system.

The component is configurable for up to 33 channel interfaces. If more than 33 CTIs must be connected together, then CTMs can be connected together without limitation.

The following figure shows the external connections on the Cross Trigger Matrix.

Figure 7-2: `css600_ctm` logical connections



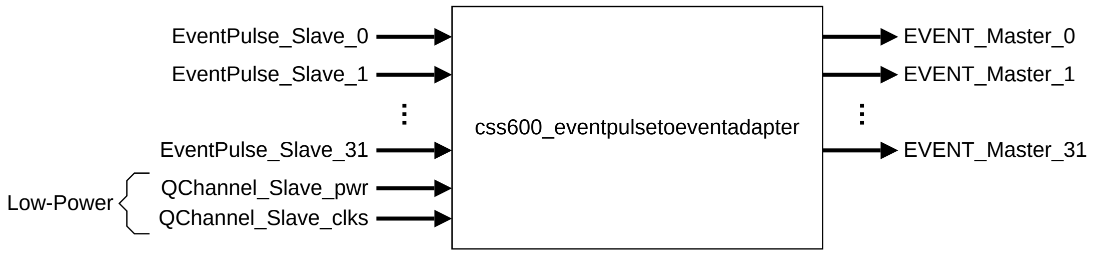
7.5 Event Pulse to Event adapter

The `Css600_eventpulsetoeventadapter` Event Pulse to Event adapter is a wrapper component that instantiates a slave half of a pulse async bridge with a configurable signal width.

The component provides the req/ack handshake that is required to interface a SoC-600M event to a legacy CTI, such as one in a CoreSight™ SoC-400M system.

The following figure shows the external connections on the Event Pulse to Event adapter.

Figure 7-3: `css600_eventpulsetoeventadapter` logical connections



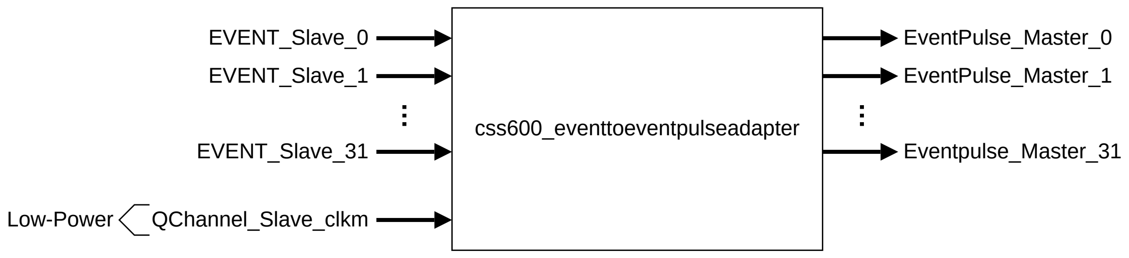
7.6 Event to Event Pulse adapter

The `css600_eventtoeventpulseadapter` Event to Event Pulse adapter is a wrapper component that instantiates a master half of a pulse async bridge with a configurable signal width.

The component provides the req/ack handshake that is required to interface a legacy event source to a SoC-600M CTI.

The following figure shows the external connections on the Event to Event Pulse adapter.

Figure 7-4: `css600_eventtoeventpulseadapter` logical connections



7.7 Event Level asynchronous bridge

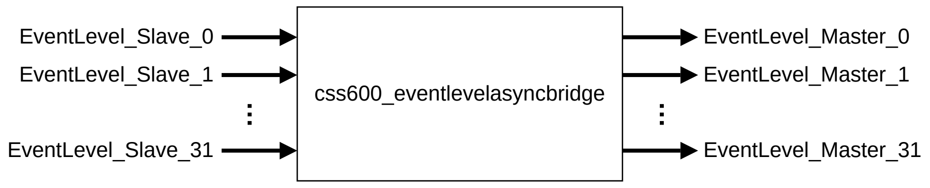
The `css600_eventlevelasyncbridge` Event Level asynchronous bridge is a wrapper component that instantiates a synchronizer.

The bridge is used to pass an event that operates as a level, rather than a pulse, across a clock domain boundary. The bridge can be used, for example, when using the software handshake configuration of a CTI event output, and the resulting event output must cross a clock or power domain boundary to reach its destination.

If more than one signal is to be transported across the same boundary, then the component can be configured for width.

The following figure shows the external connections on the Event Level asynchronous bridge.

Figure 7-5: `css600_eventlevelasyncbridge` logical connections



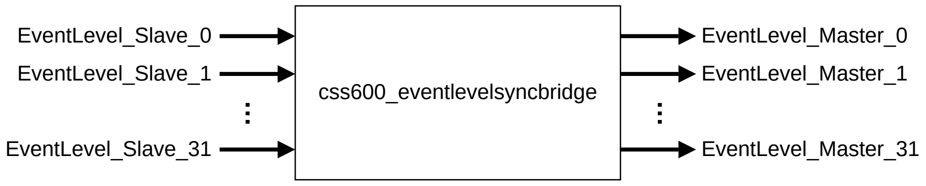
7.8 Event Level synchronous bridge

The `css600_eventlevelsyncbridge` Event Level synchronous bridge is a wrapper component that instantiates a register slice as an aid to timing closure on events that must travel a long distance on chip.

If more than one signal is to be transported across the same boundary, then the component is configurable for width.

The following figure shows the external connections on the Event Level synchronous bridge.

Figure 7-6: `css600_eventlevelsyncbridge` logical connections



7.9 Event Pulse asynchronous bridge

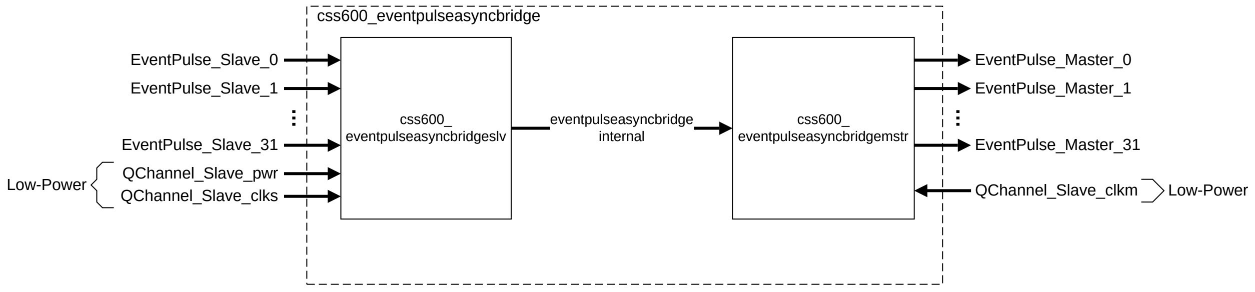
The `css600_eventpulseasyncbridge` Event Pulse asynchronous bridge is used where an event signal, or a group of events, must cross a clock or power domain boundary.

The bridge has the following features:

- Two independent clock domains with any phase or frequency alignment
- Two independent power domains, either of which can be switched relative to the other
- Three Q-Channel LPIs for slave side clock, master side clock, and power switching management
- Two-part meta-component with separate slave and master side components
- Configurable up to 32-bits wide for transporting multiple events across the same boundary

The following figure shows the external connections on the Event Pulse asynchronous bridge.

Figure 7-7: `css600_eventpulseasyncbridge` logical connections



7.10 Event Pulse synchronous bridge

The `css600_eventpulsesyncbridge` Event Pulse synchronous bridge is used where an event signal, or a group of events, must cross a synchronous clock domain boundary.

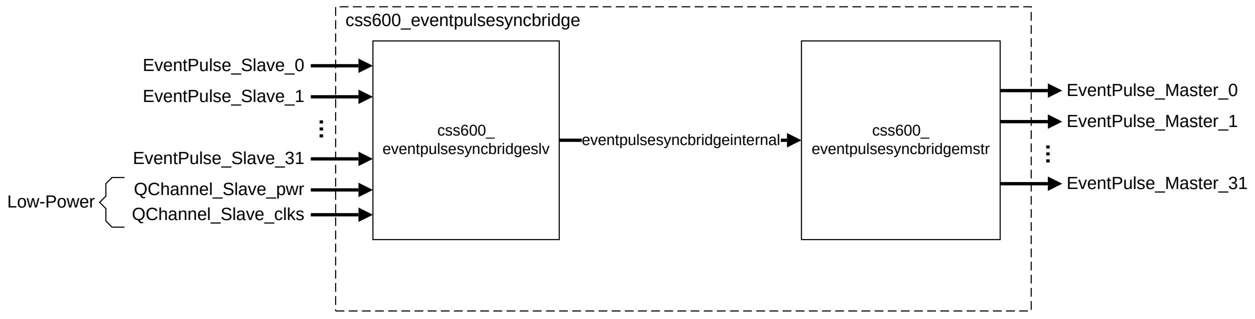
The bridge has the following features:

- Two synchronous clock domains with any frequency difference. The clocks must be skew balanced and from a common source so that they are high-level-gated by a common control point.

- Two Q-Channel LPIs for clock and power switching management
- Two-part meta-component with separate slave and master side components
- Configurable up to 32 bits wide for transporting multiple events across the same boundary

The following figure shows the external connections on the Event Pulse synchronous bridge.

Figure 7-8: `css600_eventpulsesyncbridge` logical connections



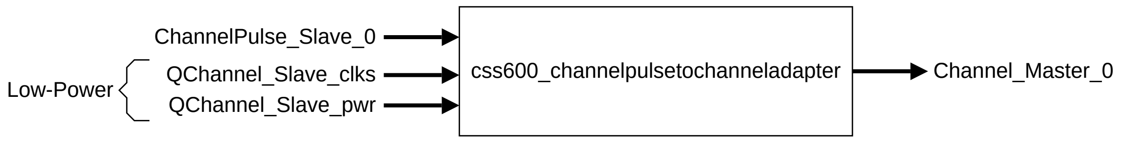
7.11 Channel Pulse to Channel adapter

The `css600_channelpulsetochanneladapter` Channel Pulse to Channel adapter is a wrapper component that instantiates a slave half of a pulse async bridge with a 4-bit signal width.

The component provides the req/ack handshake that is required to interface a SoC-600M CTI or CTM to a legacy CTI or CTM such as one in a CoreSight™ SoC-400M system.

The following figure shows the external connections on the Channel Pulse to Channel adapter.

Figure 7-9: `css600_channelpulsetochanneladapter` logical connections



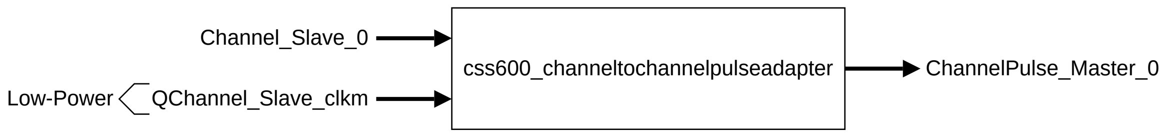
7.12 Channel to Channel Pulse adapter

The `css600_channeltochannelpulseadapter` Channel to Channel Pulse adapter is a wrapper component that instantiates a master half of a pulse async bridge with a 4-bit signal width.

The component provides the req/ack handshake that is required to interface a SoC-600M CTI or CTM to a legacy CTI or CTM such as one in a CoreSight™ SoC-400M system.

The following figure shows the external connections on the Channel to Channel Pulse adapter.

Figure 7-10: `css600_channeltochannelpulseadapter` logical connections



7.13 Channel Pulse asynchronous bridge

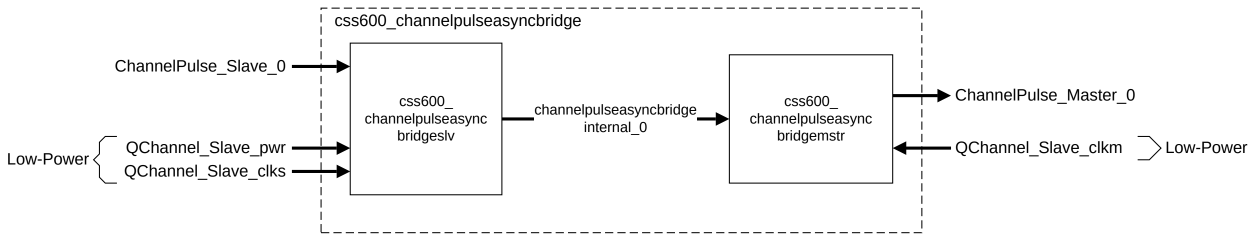
The `css600_channelpulseasyncbridge` Channel Pulse asynchronous bridge is a wrapper component that instantiates a pulse asynchronous bridge with a 4-bit signal path.

The bridge is used to connect a CTI to a CTM, or two CTMs, where the signals must cross a clock or power domain boundary. The bridge has the following features:

- Two independent clock domains with any phase or frequency alignment
- Two independent power domains, either of which can be switched relative to the other
- Two-part meta-component with separate slave and master side components

The following figure shows the external connections on the Channel Pulse asynchronous bridge.

Figure 7-11: `css600_channelpulseasyncbridge` logical connections


css600_channelpulseasyncbridge
ChannelPulse_Slave_0
Low-Power
QChannel_Slave_pwr
QChannel_Slave_clks
css600_ channelpulseasync bridgeslv
channelpulseasyncbridge internal_0
css600_ channelpulseasync bridgemstr
ChannelPulse_Master_0
QChannel_Slave_clkm
Low-Power

7.14 Channel Pulse synchronous bridge

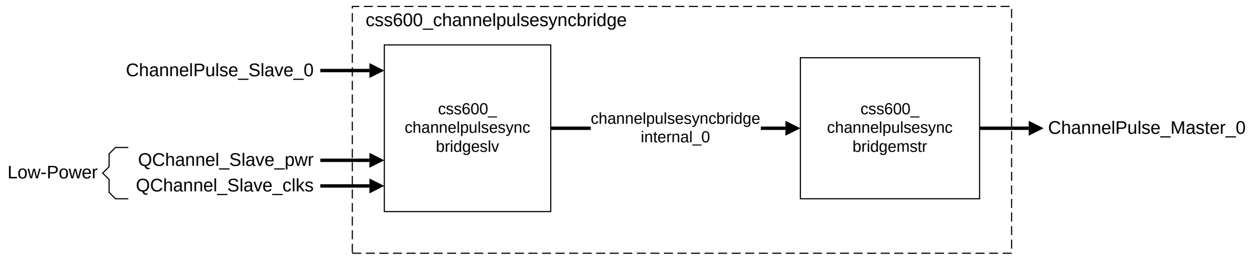
The `css600_channelpulsesyncbridge` Channel Pulse synchronous bridge is a wrapper component that instantiates a pulse synchronous bridge with a 4-bit signal path.

The bridge is used to connect a CTI to a CTM, or two CTMs, where the signals must cross a synchronous clock domain boundary. The bridge has the following features:

- Two synchronous clock domains with any frequency difference. The clocks must be skew balanced, and from a common source, so that they are high-level-gated by a common control point.
- Two Q-Channel LPIs for clock and power switching management
- Two-part meta-component with separate slave and master side components

The following figure shows the external connections on the Channel Pulse synchronous bridge.

Figure 7-12: `css600_channelpulsesyncbridge` logical connections


css600_channelpulsesyncbridge
ChannelPulse_Slave_0
Low-Power
QChannel_Slave_pwr
QChannel_Slave_clks
css600_ channelpulsesync bridgeslv
channelpulsesyncbridge internal_0
css600_ channelpulsesync bridgemstr
ChannelPulse_Master_0

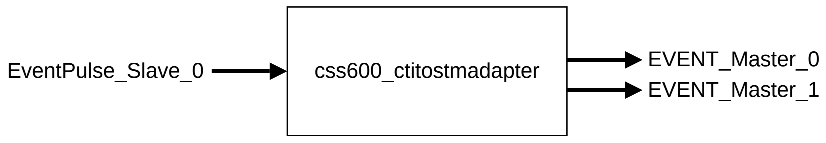
7.15 CTI to STM adapter

The `css600_ctitostmadapter` is an IP-XACT phantom component that is provided to support stitching in an IP-XACT tooling product. There is no Verilog module for `css600_ctitostmadapter`.

The `css600_ctitostmadapter` CTI to STM adapter is used to adapt a single event signal to two event inputs of a System Trace Macrocell.

The following figure shows the external connections on the CTI to STM adapter.

Figure 7-13: `css600_ctitostmadapter` logical connections



8. Authentication components functional description

This chapter describes the functionality of the authentication components.

8.1 Authentication replicator

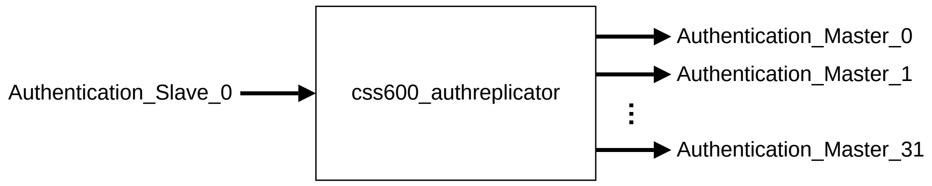
The `css600_authreplicator` is an IP-XACT phantom component that is provided to support stitching in an IP-XACT tooling product.

There is no Verilog module for `css600_authreplicator`.

Use the `css600_authreplicator` to connect a single Authentication master interface to multiple Authentication slave interfaces.

The following figure shows the external connections on the Authentication replicator.

Figure 8-1: `css600_authreplicator` logical connections



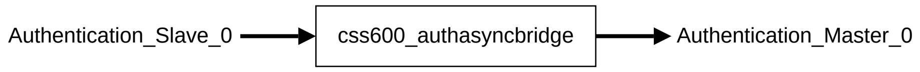
8.2 Authentication asynchronous bridge

The `css600_authasyncbridge` authentication asynchronous bridge is used where the Authentication interface must cross a clock or power domain boundary.

The bridge contains synchronizers to capture the signals in the receiving clock domain.

The following figure shows the external connections on the Authentication asynchronous bridge.

Figure 8-2: `css600_authasyncbridge` logical connections

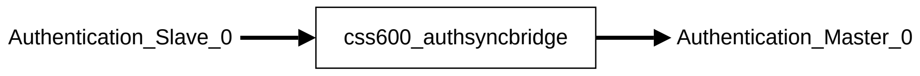


8.3 Authentication synchronous bridge

The `css600_authsyncbridge` authentication synchronous bridge is a register slice to aid timing closure for authentication signals that are crossing a large distance across a chip.

The following figure shows the external connections on the Authentication synchronous bridge.

Figure 8-3: `css600_authsyncbridge` logical connections



9. Processor Integration Layer components

This chapter gives an overview of the Cortex® *Processor Integration Layers* (PILs).

9.1 Cortex-M0 PIL overview

The Cortex®-M0 *Processor Integration Layer* (PIL) consists of the following:

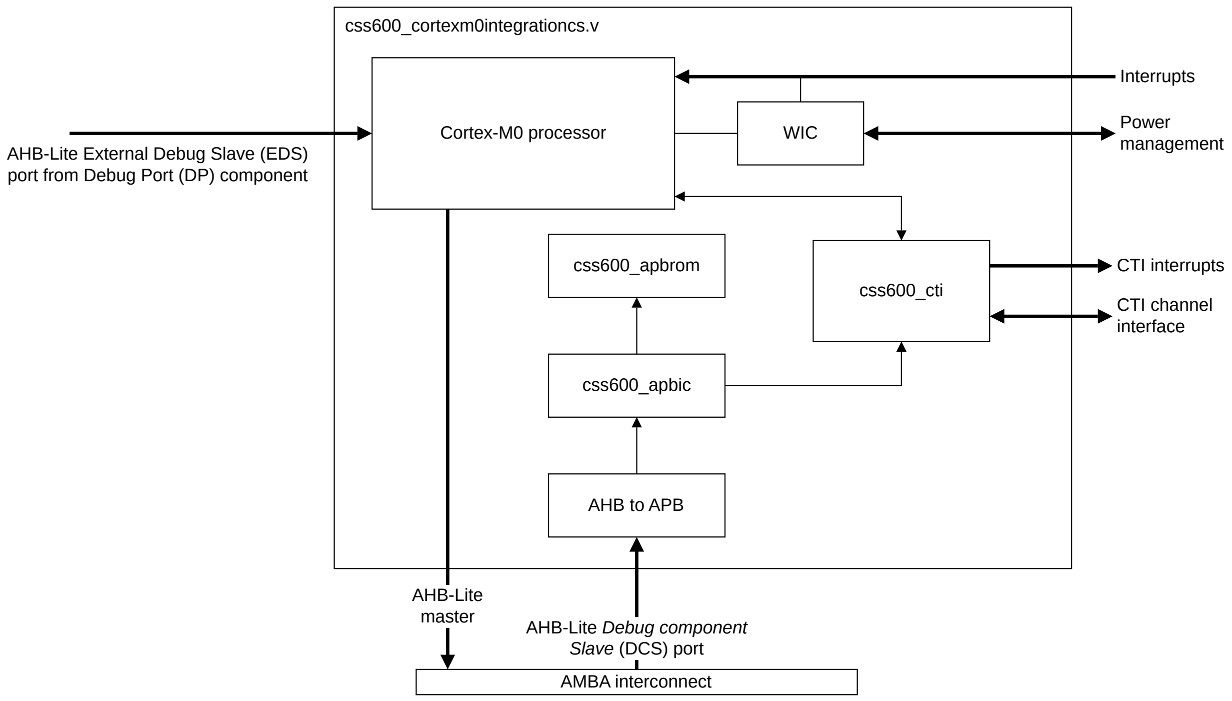
- A Cortex®-M0 processor
- An optional *Wake up Interrupt Controller* (WIC)
- A ROM table to identify the PIL contents
- A *Cross Trigger Interface* (CTI) for debug event communication

The Cortex®-M0 PIL has the following interfaces:

- An AHB-Lite master interface that connects to the system *Network Interconnect* (NIC)
- An AHB slave interface that connects to the AHB-AP port of the CoreSight™ DAP
- An AHB slave interface for accessing the CTI and ROM table
- Processor-specific signals such as interrupt signals, system control signals, and status signals

The following figure shows a block diagram of the Cortex®-M0.

Figure 9-1: Cortex®-M0 PIL block diagram



9.1.1 Cortex-M0 PIL CoreSight component identification

CoreSight™ components have several IDs that identify the components.

The following table shows the CoreSight™ ID register reset values for the components present within the Cortex®-M0 PIL. See the *Arm® CoreSight™ Architecture Specification v3.0* for information on the CoreSight™ ID scheme.

Table 9-1: Cortex®-M0 PIL CoreSight™ ID register reset values

PID	CID	DevType	DevArch	Revision	Component
`0x00000004001BB4C2`	`0xB105900D`	`0x00`	`0x47700AF7`	r0p0	`css600_cortexm0integrationcs` ROM Table
`0x00000004000BB008`	`0xB105E00D`	`0x00`	`0x00000000`	r0p0	Armv6M *System Control Space* (SCS)
`0x00000004000BB00A`	`0xB105E00D`	`0x00`	`0x00000000`	r0p0	Armv6M *Data Watchpoint and Trace* (DWT)
`0x00000004000BB00B`	`0xB105E00D`	`0x00`	`0x00000000`	r0p0	Armv6M *FlashPatch and Breakpoint* (FPB)
`0x00000004003BB9ED`	`0xB105900D`	`0x14`	`0x47701A14`	r0p3	`css600_cti`

9.1.2 Cortex-M0 PIL Debug memory map

The debug components in the Cortex®-M0 PIL share memory space with the processor system.

You must build your system level interconnect so that the PIL *Debug Component Slave* (DCS) AHB-Lite port is accessed for the address ranges of the PIL components.

The following table shows the locations of the Cortex®-M0 PIL CoreSight™ components.

Table 9-2: Cortex®-M0 PIL debug memory map

Address range	Components
`0xF0000000-0xF0000FFF`	PIL primary ROM table
`0xF0001000-0xF0001FFF`	CTI

9.2 Cortex-M3 PIL overview

The Cortex®-M3 *Processor Integration Layer* (PIL) consists of the following:

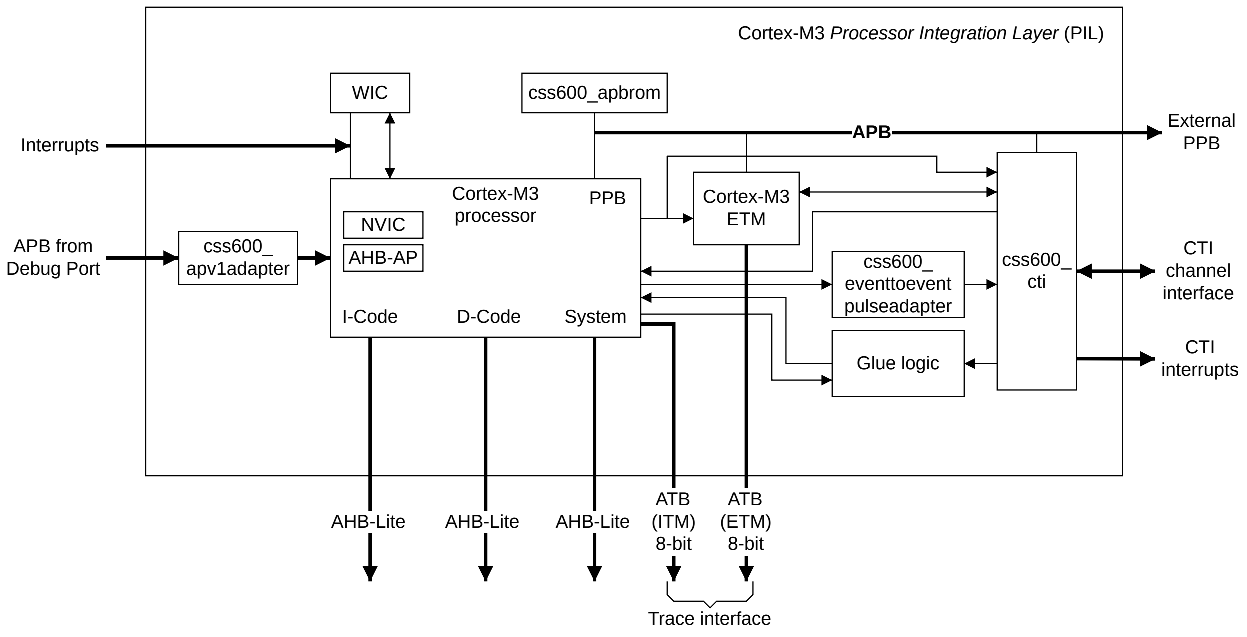
- A processor that has an *Instrumentation Trace Macrocell* (ITM) and AHB-(AP)
- An optional *Wakeup Interrupt Controller* (WIC)
- A ROM table that connects to the processor through a *Private Peripheral Bus* (PPB)
- An ETM trace unit that connects to the processor
- A CTI for debug event communication

The Cortex®-M3 PIL supports the following external interfaces:

- AHB-Lite interfaces:
    - I-Code
    - D-Code
    - System
- Two ATB interfaces that connect to the CoreSight™ subsystem
- An APB interface for adding debug components to the PPB
- An APB interface that connects to the debug port in the CoreSight™ subsystem
- Processor-specific signals such as interrupt signals, system control signals, and status signals

The following figure shows a block diagram of the Cortex®-M3 PIL.

Figure 9-2: Cortex®-M3 PIL block diagram



9.2.1 Cortex-M3 PIL CoreSight component identification

CoreSight™ components have several IDs that identify the components.

The following table shows the CoreSight™ ID register reset values for the components present within the Cortex®-M3 PIL. See the *Arm® CoreSight™ Architecture Specification v3.0* for information on the CoreSight™ ID scheme.

Table 9-3: Cortex®-M3 PIL CoreSight™ ID register reset values

PID	CID	DevType	DevArch	Revision	Component
`0x00000004000BB9E5`	`0xB105900D`	`0x00`	`0x47700A47`	r0p0	`css600_apv1adapter`
`0x00000004001BB4C5`	`0xB105900D`	`0x00`	`0x47700AF7`	r0p0	`css600_cortexm3integrationcs` ROM Table

PID	CID	DevType	DevArch	Revision	Component
`0x00000004000BB000`	`0xB105E00D`	`0x00`	`0x00000000`	r0p0	Armv7M *System Control Space* (SCS)
`0x00000004003BB002`	`0xB105E00D`	`0x00`	`0x00000000`	r0p0	Arm v7M *Data Watchpoint and Trace* (DWT)
`0x00000004002BB003`	`0xB105E00D`	`0x00`	`0x00000000`	r0p0	Arm v7M *FlashPatch and Breakpoint* (FPB)
`0x00000004003BB001`	`0xB105E00D`	`0x00`	`0x00000000`	r0p0	Armv7M *Instrumentation Trace Macrocell* (ITM)
`0x00000004003BB924`	`0xB105900D`	`0x13`	`0x00000000`	r0p0	Cortex®-M3 ETM
`0x00000004003BB9ED`	`0xB105900D`	`0x14`	`0x47701A14`	r0p3	`css600_cti`

9.2.2 Cortex-M3 PIL Debug memory map

The debug components in the Cortex®-M3 PIL share memory space with the processor system. Part of the system memory is allocated to the *Private Peripheral Bus* (PPB).

The following tables show the locations of the Cortex®-M3 PIL CoreSight™ components.

Table 9-4: External PPB division

Address range	Components
`0xE0041000-0xE0041FFF`	ETM trace unit
`0xE0042000-0xE0042FFF`	CTI
`0xE00FF000-0xE00FFFFF`	ROM table
`0xE0040000-0xE0040FFF`	External PPB expansion bus. In a standard single processor Cortex®-M3 system, the Cortex®-M3 TPIU uses this space
`0xE0043000-0xE00FEFFF`	External PPB expansion bus

Table 9-5: Internal PPB division

Address range	Section	Components
`0xE0000000-0xE003FFFF`	Internal PPB	These components are: • *Instrumentation Trace Macrocell* (ITM) • *Data Watchpoint and Trace* (DWT) • *Flash Patch and Breakpoint* (FPB) • *System Control Space* (SCS) including for example: ◦ *Nested Vectored Interrupt Controller* (NVIC) ◦ SysTick ◦ *Memory Protection Unit* (MPU)
`0xE0040000-0xE00FFFFF`	External PPB	These components are: • ROM table • *Embedded Trace Macrocel* (ETM) trace unit • *Cross Trigger Interface* (CTI)

9.3 Cortex-M4 PIL overview

The Cortex®-M4 *Processor Integration Layer* (PIL) consists of the following:

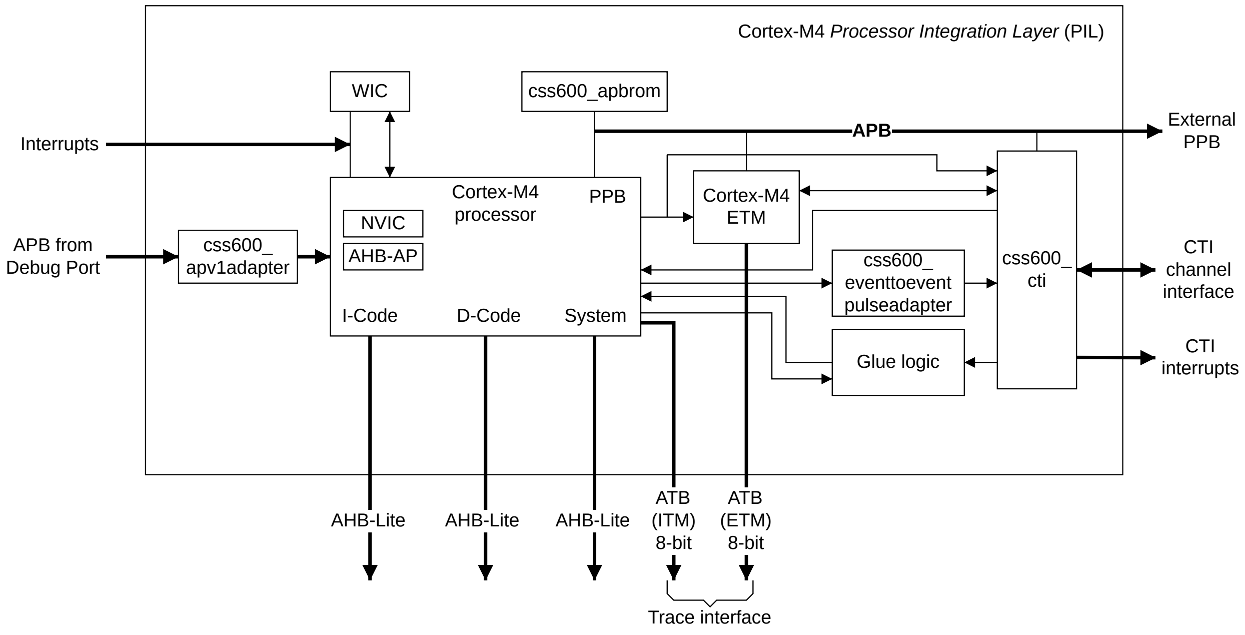
- A processor that has an *Instrumentation Trace Macrocell* (ITM) and *Advanced High-performance Bus* (AHB)-*Access Port* (AP)
- An optional *Wakeup Interrupt Controller* (WIC)
- A ROM table that connects to the processor through a *Private Peripheral Bus* (PPB)
- An *Embedded Trace Macrocell* (ETM) trace unit that connects to the processor
- A CTI for debug event communication

The Cortex®-M4 PIL supports the following external interfaces:

- AHB-Lite interfaces:
  - I-Code
  - D-Code
  - System
- Two *Advanced Trace Bus* (ATB) interfaces that connect to the CoreSight™ subsystem
- An *Advanced Peripheral Bus* (APB) interface for adding debug components to the PPB
- An APB interface that connects to the debug port in the CoreSight™ subsystem
- Processor-specific signals such as interrupt signals, system control signals, and status signals

The following figure shows a block diagram of the Cortex®-M4 PIL.

Figure 9-3: Cortex®-M4 PIL block diagram



9.3.1 Cortex-M4 PIL CoreSight component identification

CoreSight™ components have several IDs that identify the components.

The following table shows the CoreSight™ ID register reset values for the components present within the Cortex®-M4 PIL. See the *Arm® CoreSight™ Architecture Specification v3.0* for information on the CoreSight™ ID scheme.

Table 9-6: Cortex®-M4 PIL CoreSight™ ID register reset values

PID	CID	DevType	DevArch	Revision	Component
`0x00000004000BB9E5`	`0xB105900D`	`0x00`	`0x47700A47`	r0p0	`css600_apv1adapter`
`0x00000004001BB4C6`	`0xB105900D`	`0x00`	`0x47700AF7`	r0p0	`css600_cortexm4integrationcs` ROM table
`0x00000004000BB00C`	`0xB105E00D`	`0x00`	`0x00000000`	r0p0	Armv7M *System Control Space* (SCS)
`0x00000004003BB002`	`0xB105E00D`	`0x00`	`0x00000000`	r0p0	Armv7M *Data Watchpoint and Trace* (DWT)
`0x00000004002BB003`	`0xB105E00D`	`0x00`	`0x00000000`	r0p0	Armv7M *FlashPatch and Breakpoint* (FPB)
`0x00000004003BB001`	`0xB105E00D`	`0x00`	`0x00000000`	r0p0	Armv7M *Instrumentation Trace Macrocell* (ITM)
`0x00000004000BB925`	`0xB105900D`	`0x13`	`0x00000000`	r0p0	Cortex®-M4 ETM
`0x00000004003BB9ED`	`0xB105900D`	`0x14`	`0x47701A14`	r0p3	`css600_cti`

9.3.2 Cortex-M4 PIL Debug memory map

The debug components in the Cortex®-M4 PIL share memory space with the processor system. Part of the system memory is allocated to the *Private Peripheral Bus* (PPB).

The following tables show the locations of the Cortex®-M4 PIL CoreSight™ components.

Table 9-7: External PPB division

Address range	Components
`0xE0041000-0xE0041FFF`	ETM trace unit
`0xE0042000-0xE0042FFF`	CTI
`0xE00FF000-0xE00FFFFF`	ROM table
`0xE0040000-0xE0040FFF`	External PPB expansion bus. In a standard single processor Cortex®-M4 system, the Cortex®-M4 TPIU uses this space
`0xE0043000-0xE00FEFFF`	External PPB expansion bus

Table 9-8: Internal PPB division

Address range	Section	Components
`0xE0000000-0xE003FFFF`	Internal PPB	These components are: • *Instrumentation Trace Macrocell* (ITM) • *Data Watchpoint and Trace* (DWT) • *Flash Patch and Breakpoint* (FPB) • *System Control Space* (SCS) including for example: ◦ *Nested Vectored Interrupt Controller* (NVIC) ◦ SysTick ◦ *Memory Protection Unit* (MPU)
`0xE0040000-0xE00FFFFF`	External PPB	These components are: • ROM table • *Embedded Trace Macrocell* (ETM) trace unit • *Cross Trigger Interface* (CTI)

10. Programmers model

This chapter describes the programmers models for all CoreSight™ SoC-600M components that have programmable registers.

10.1 Components programmers model

The following information applies to the SoC-600M components registers:

- The base address of any component is not fixed, and can be different for any particular system implementation. The offset of each register within a component from the component base address is fixed.
- Do not attempt to access reserved or unused address locations. Attempting to access these locations can result in **UNPREDICTABLE** behavior.
- Unless otherwise stated in the accompanying text:
    - Do not modify undefined register bits.
    - Ignore undefined register bits on reads.
    - All register bits are reset to the reset value specified in the register summary table for the component.
- Access types are described as follows:

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

10.2 `css600_dp` introduction

This section describes the programmers model of the `css600_dp`.

The register block in the SoC-600M DP is shared between two different protocol engines, the JTAG-DP and the SW-DP.

The DP programmers model consists of the following registers. The programmers model is based on the *Arm® Debug Interface Architecture Specification ADIv6.0*. Because the DP only supports 32-bit addressing, any read to BASEPTR1 always returns 0 and any writes to SELECT1 register are ignored.

10.2.1 `css600_dp` register summary

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

More than one register can appear at a given address, depending on the value of SELECT.DPBANKSEL. The combinations of address offset and SELECT.DPBANKSEL value, and whether the register is accessible by the JTAG-DP, SW-DP, or both, are all shown in the following table.

Note

A reset value containing one or more '-' means that this register contains **UNKNOWN** or **IMPLEMENTATION-DEFINED** values. See the relevant register description for more information.

A DPBANKSEL value containing an 'X' means that the DPBANKSEL value is ignored.

Locations that are not listed in the table are Reserved.

Table 10-1: `css600_dp` register summary

Offset	DPBANKSEL	Name	JTAG-DP	SW-DP	Reset	Description
-	X	IDCODE	No	No	`0x4BA06477` or `0x4BA07477`	css600_dp JTAG TAP ID Register, IDCODE
-	X	ABORT	No	Yes	`0x00000000`	css600_dp AP Abort Register, ABORT
`0x000000--`	`0x0`	DPIDR	Yes	Yes	`0x4C013477`	css600_dp Debug Port Identification Register, DPIDR
`0x0000`	`0x1`	DPIDR1	Yes	Yes	`0x000000--`	css600_dp Debug Port Identification Register 1, DPIDR1
`0x00000000`	`0x2`	BASEPTR0	Yes	Yes	`0x-----00-`	css600_dp Base Pointer Register 0, BASEPTR0
`0x0000`	`0x3`	BASEPTR1	Yes	Yes	`0x00000000`	css600_dp Base Pointer Register 1, BASEPTR1
`0x0004`	`0x0`	CTRLSTAT	Yes	Yes	`0x--000000`	css600_dp Control/Status Register, CTRLSTAT
`0x0004`	`0x1`	DLCR	No	Yes	`0x00000040`	css600_dp Data Link Control Register, DLCR
`0x0004`	`0x2`	TARGETID	Yes	Yes	`0x--------`	css600_dp Target Identification Register, TARGETID
`0x0004`	`0x3`	DLPIDR	Yes	Yes	`0x-0000001`	css600_dp Data Link Protocol Identification Register, DLPIDR
`0x0004`	`0x4`	EVENTSTAT	Yes	Yes	`0x0000000-`	css600_dp Event Status Register, EVENTSTAT
`0x0004`	`0x5`	SELECT1	Yes	Yes	`0x00000000`	css600_dp Select Register 1, SELECT1
`0x0008`	X on reads	RESEND	No	Yes	`0x00000000`	css600_dp Read Resend Register, RESEND
`0x0008`	X on writes	SELECT	Yes	Yes	`0x00000000`	css600_dp Select Register, SELECT
`0x000C`	X on reads	RDBUFF	No	Yes	`0x00000000`	css600_dp Read Buffer Register, RDBUFF
`0x000C`	X on writes	TARGETSEL	No	Yes	`0x00000000`	css600_dp Target Selection Register, TARGETSEL

10.2.2 `css600_dp` register descriptions

This section describes the `css600_dp` registers.

10.2.1 css600_dp register summary on page 88 provides cross references to individual registers.

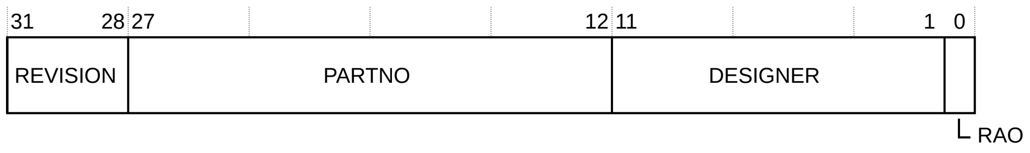
10.2.2.1 `css600_dp` JTAG TAP ID Register, IDCODE

The IDCODE value enables a debugger to identify the JTAG DP to which it is connected.

JTAG-DP Access is through its own scan-chain using the IDCODE instruction in the JTAG IR. SW-DP There is no IDCODE register in the SW-DP.

The following figure shows the bit assignments.

Figure 10-1: IDCODE register bit assignments



The following table shows the bit assignments.

Table 10-2: IDCODE register bit assignments

Bits	Reset value	Name	Function
[31:28]	`0x4`	REVISION	Revision. An incremental value starting at `0x0` for the first design of a component. See the Component list in Chapter 1 for information on the RTL revision of the component.
[27:20]	IMPLEMENTATION DEFINED	PARTNO	Part Number of the DP. The value depends on the Instruction Register length configuration of the `css600_dp`: **`0xBA06`** 4-bit IR **`0xBA07`** 8-bit IR
[11:1]	`0x23B`	DESIGNER	Designer ID based on 11-bit JEDEC JEP106 continuation and identity code `0x23B`, Arm Ltd
[0]	`0b1`	RAO	RAO

See Arm® *Debug Interface Architecture Specification ADIv6.0* for details about accessing the IDCODE value in a JTAG DP.

10.2.2.2 `css600_dp` AP Abort Register, ABORT

The ABORT register drives the dp_abort pin on the DP, which goes to APs to abort the current transaction.

JTAG-DP Access is through its own scan-chain using the ABORT instruction in the JTAG IR.

SW-DP Access is by a write to offset `0x0` of the DP register map.

Attributes

Offset

`0x0000`

Type

Write-only

Reset

`0x00000000`

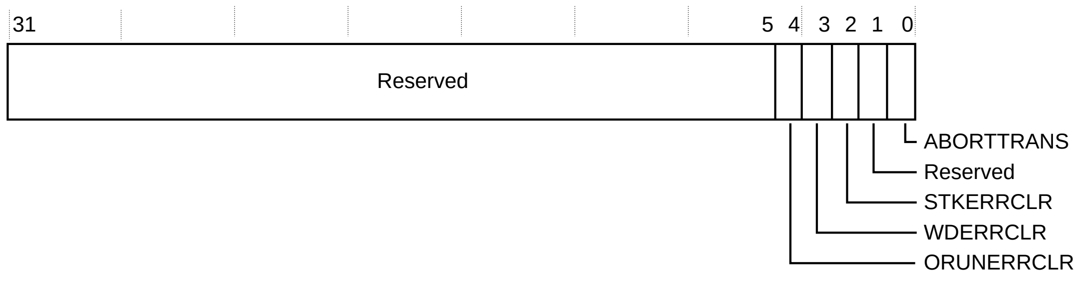
Width

32

Bit descriptions

The following figure shows the bit assignments.

Figure 10-2: ABORT register bit assignments



The following table shows the bit assignments.

Table 10-3: ABORT register bit assignments

Bits	Reset value	Name	Function
[4]	`0b0`	ORUNERRCLR	Write 1 to this bit to clear the CTRLSTAT.STICKYORUN overrun error bit to 0
[3]	`0b0`	WDERRCLR	Write 1 to this bit to clear the CTRLSTAT.WDATAERR write data error bit to 0
[2]	`0b0`	STKERRCLR	Write 1 to this bit to clear the CTRLSTAT.STICKYERR sticky error bit to 0
[0]	`0b0`	ABORTTRANS	Write 1 to this bit to generate an abort. This aborts the current AP transaction

10.2.2.3 `css600_dp` Debug Port Identification Register, DPIDR

The DPIDR provides information about the DP.

Attributes

Offset

`0x0000`

Type

Read-only

Reset

`0x4C013477`

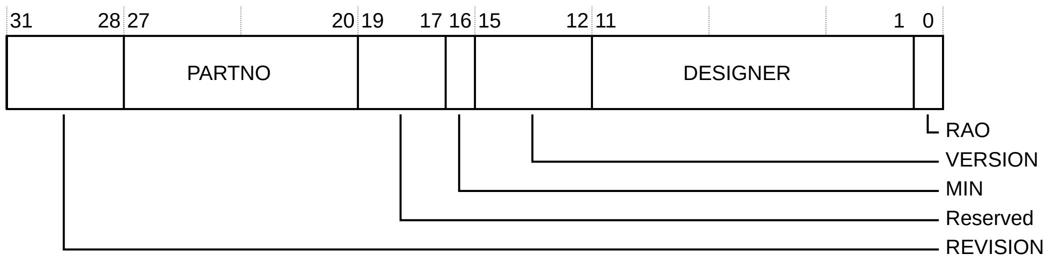
Width

32

Bit descriptions

The following figure shows the bit assignments.

Figure 10-3: DPIDR register bit assignments



The following table shows the bit assignments.

Table 10-4: DPIDR register bit assignments

Bits	Reset value	Name	Function
[31:28]	`0b0100`	REVISION	Revision code: `0b0100` - r0p4
[27:20]	`0b11000000`	PARTNO	Part Number of the DP
[16]	`0b1`	MIN	Transaction counter, Pushed-verify, and Pushed-find operations are not implemented
[15:12]	`0b0011`	VERSION	Version of DP architecture implemented: SoC-600M is DPv3, so the value of this field is `0x3`
[11:1]	`0b01000111011`	DESIGNER	Designer ID based on 11-bit JEDEC JEP106 continuation and identity code: the Arm value is `0x23B` for this field
[0]	`0b1`	RAO	RAO

10.2.2.4 `css600_dp` Debug Port Identification Register 1, DPIDR1

The DPIDR1 register is the extension of DPIDR and provides information about the DP.

Attributes

Offset

`0x0000`

Type

Read-only

Reset

`0x000000--`

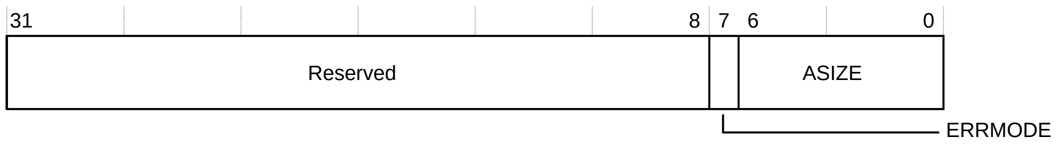
Width

32

Bit descriptions

The following figure shows the bit assignments.

Figure 10-4: DPIDR1 register bit assignments



The following table shows the bit assignments.

Table 10-5: DPIDR1 register bit assignments

Bits	Reset value	Name	Function
[7]	`0b1`	ERRMODE	Error reporting mode support: **1** CTRLSTAT.ERRMODE implemented.
[6:0]	IMPLEMENTATION DEFINED	ASIZE	Address size. This defines the size of the address in the SELECT register, and the BASEPTR0 register. Allowed values are: **`0x0C`** 12-bit address **`0x14`** 20-bit address **`0x20`** 32-bit address All other values are reserved. This is an IMPLEMENTATION-DEFINED value that depends on the configuration of the component.

10.2.2.5 `css600_dp` Base Pointer Register 0, BASEPTR0

BASEPTR0 and BASEPTR1 together provide an initial system address for the first component in the system. Typically, this is the address of a top-level ROM table that indicates where APv2 APs are located. The size of the address is defined in DPIDR1.ASIZE, which defines the size of the whole address even though bits [11:0] are always zero, as the minimum address space for each component is 4KB.

Attributes

Offset

`0x-----00-`

Type

Read-only

Reset

`0x0000000-`

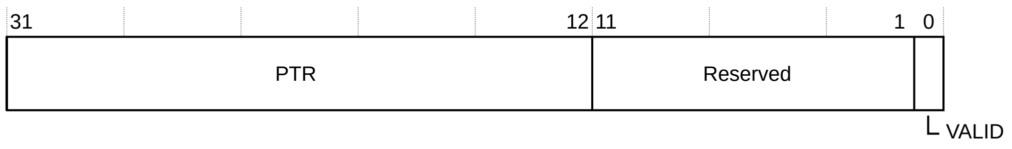
Width

32

Bit descriptions

The following figure shows the bit assignments.

Figure 10-5: BASEPTR0 register bit assignments



The following table shows the bit assignments.

Table 10-6: BASEPTR0 register bit assignments

Bits	Reset value	Name	Function
[31:12]	IMPLEMENTATION DEFINED	PTR	Base address bits [31:12] of first component in the system. The address is aligned to a 4KB boundary. This IMPLEMENTATION-DEFINED value depends on the interface tie-off value of baseaddr.
[0]	IMPLEMENTATION DEFINED	VALID	Indicates whether the base address is valid. Depends on the interface tie-off value of baseaddr_valid. **0** No base address specified. PTR is UNKNOWN. **1** Base address is specified in PTR.

10.2.2.6 `css600_dp` Base Pointer Register 1, BASEPTR1

Because this product supports 32-bit addressing only, BASEPTR1 always reads 0s.

Attributes

Offset

`0x0000`

Type

Read-only

Reset

`0x00000000`

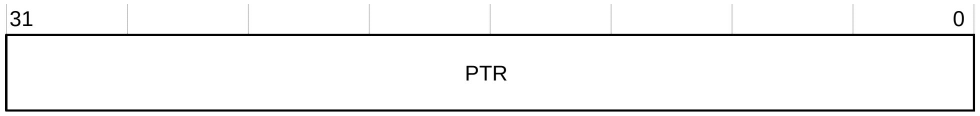
Width

32

Bit descriptions

The following figure shows the bit assignments.

Figure 10-6: BASEPTR1 register bit assignments



The following table shows the bit assignments.

Table 10-7: BASEPTR1 register bit assignments

Bits	Reset value	Name	Function
[31:0]	`0x0`	PTR	Base address bits [63:32] of first component in the system. Always reads 0s.

10.2.2.7 `css600_dp` Control/Status Register, CTRLSTAT

The Control/Status register provides control of the DP and status information about the DP.

Attributes

Offset

`0x0004`

Type

Read-write

Reset

`0x--000000`

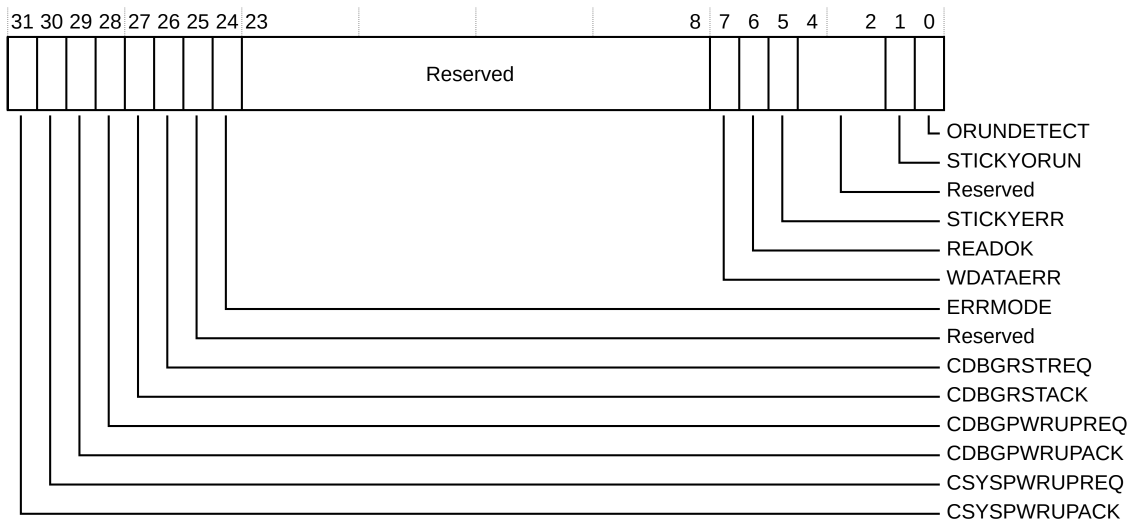
Width

32

Bit descriptions

The following figure shows the bit assignments.

Figure 10-7: CTRLSTAT register bit assignments



The following table shows the bit assignments.

Table 10-8: CTRLSTAT register bit assignments

Bits	Reset value	Name	Function
[31]	UNKNOWN	CSYSPWRUPACK	System powerup acknowledge. Status of CSYSPWRUPACK interface signal.
[30]	`0b0`	CSYSPWRUPREQ	System powerup request. This bit controls the CSYSPWRUPREQ signal on the interface.
[29]	UNKNOWN	CDBGPWRUPACK	Debug powerup acknowledge. Status of CDBGPWRUPACK interface signal.
[28]	`0b0`	CDBGPWRUPREQ	Debug powerup request. This bit controls the CDBGPRWUPREQ signal on the interface.
[27]	UNKNOWN	CDBGRSTACK	Debug reset acknowledge. Indicates the status of the CDBGRSTACK signal on the interface.
[26]	`0b0`	CDBGRSTREQ	Debug reset request. This bit controls the CDBGRSTREQ signal on interface.
[24]	`0b0`	ERRMODE	Error Mode. **0** Errors on AP transactions set CTRLSTAT.STICKYERR **1** Errors on AP transactions do not set CTRLSTAT.STICKYERR
[7]	`0b0`	WDATAERR	This bit is DATA LINK DEFINED, such that on a JTAG-DP this bit is reserved, RES0, and on an SW-DP this bit is RO. This bit is set to 1 if a Write Data Error occurs. This happens if there is a parity or framing error on the data phase of a write, or a write that has been accepted by the DP is then discarded without being submitted to the AP. On an SW-DP, this bit is cleared to 0 by writing 1 to the ABORT.WDERRCLR bit.
[6]	`0b0`	READOK	This flag always indicates the response to the last AP read access. This bit is DATA LINK DEFINED. On JTAG-DP, the bit is reserved, RES0, and on SW-DP, access is RO. If the response to the previous AP read or RDBUFF read was OK, then the bit is set to 1. If the response was not OK, then it is cleared to 0.

Bits	Reset value	Name	Function
[5]	`0b0`	STICKYERR	If an error is returned by an AP transaction, and CTRLSTAT.ERRMODE is b0, then this bit is set to 1. The behavior on writing is **DATA LINK DEFINED**: On a JTAG-DP, access is R/W1C. On a SW-DP, access is RO/WI. Clearing this bit to 0 is also **DATA LINK DEFINED**: On a JTAG-DP, the bit is cleared by writing 1 to this bit, or by writing 1 to the ABORT.STKERRCLR field. On SW-DP, the bit is cleared by writing 1 to the ABORT.STKERRCLR field.
[1]	`0b0`	STICKYORUN	If overrun detection is enabled, this bit is set to 1 when an overrun occurs. The behavior on writing is **DATA LINK DEFINED**: on a JTAG-DP, access is R/W1C. On a SW-DP, access is RO/WI. Clearing this bit to 0 is also **DATA LINK DEFINED**: On a JTAG-DP, the bit is cleared by writing 1 to this bit, or by writing 1 to the ABORT.ORUNERRCLR field. On SW-DP, the bit is cleared by writing 1 to the ABORT.ORUNERRCLR field.
[0]	`0b0`	ORUNDETECT	This bit is set to 1 to enable overrun detection

10.2.2.8 `css600_dp` Data Link Control Register, DLCR

The DLCR controls the operating mode of the Data link. Access to this register is **DATA LINK DEFINED**. For JTAG-DP, this register is Reserved RES0. For SW-DP, the register is as shown in the following table.

Attributes

Offset

`0x0004`

Type

Read-write

Reset

`0x00000040`

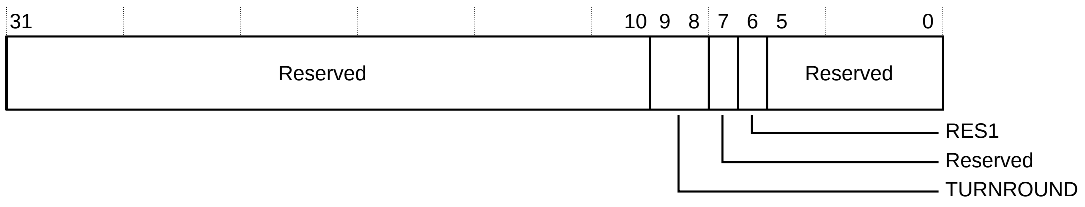
Width

32

Bit descriptions

The following figure shows the bit assignments.

Figure 10-8: DLCR register bit assignments



The following table shows the bit assignments.

Table 10-9: DLCR register bit assignments

Bits	Reset value	Name	Function
[9:8]	`0b00`	TURNROUND	Turnaround tristate period: **`0x0`** 1 data period **`0x1`** 2 data periods **`0x2`** 3 data periods **`0x3`** 4 data periods
[6]	`0b1`	RES1	Reserved, RES1

10.2.2.9 `css600_dp` Target Identification Register, TARGETID

The TARGETID register provides information about the target when the host is connected to a single device.

Attributes

Offset

`0x0004`

Type

Read-only

Reset

`0x--------`

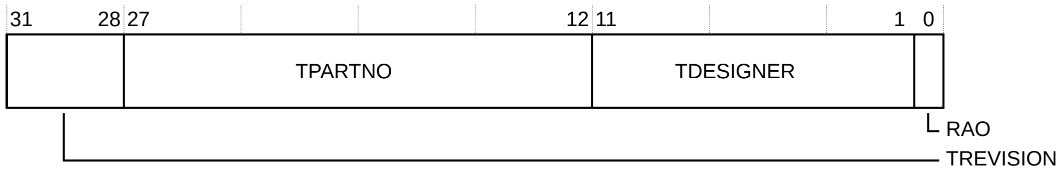
Width

32

Bit descriptions

The following figure shows the bit assignments.

Figure 10-9: TARGETID register bit assignments



The following table shows the bit assignments.

Table 10-10: TARGETID register bit assignments

Bits	Reset value	Name	Function
[31:28]	IMPLEMENTATION DEFINED	TREVISION	Target revision. The value comes from the tie-off signal targetid[31:28].
[27:12]	IMPLEMENTATION DEFINED	TPARTNO	Target part number. The value comes from the tie-off signal targetid[27:12].

Bits	Reset value	Name	Function
[11:1]	**IMPLEMENTATION DEFINED**	TDESIGNER	Designer ID, based on 11-bit JEDEC JEP106 continuation and identity code. The value comes from the tie-off signal targetid[11:1].
[0]	`0b1`	RAO	Reserved, RAO

10.2.2.10 `css600_dp` Data Link Protocol Identification Register, DLPIDR

The DLPIDR provides protocol version information. The contents of this register are **DATA LINK DEFINED**.

Attributes

Offset

`0x0004`

Type

Read-only

Reset

`0x-00000001`

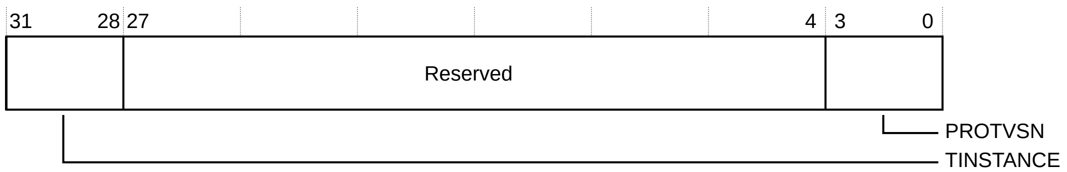
Width

32

Bit descriptions

The following figure shows the bit assignments.

Figure 10-10: DLPIDR register bit assignments



The following table shows the bit assignments.

Table 10-11: DLPIDR register bit assignments

Bits	Reset value	Name	Function
[31:28]	**IMPLEMENTATION DEFINED**	TINSTANCE	The instance number for this device. For JTAG-DP: Reserved, RES0, and for SW-DP: The value comes from the tie-off signal instanceid[3:0]. Must be unique in a multi-drop system.
[3:0]	`0b0001`	PROTVSN	Defines the protocol version that is implemented. For JTAG-DP: `0x1`, as JTAG protocol version 1 is implemented, and for SW-DP: `0x1` as SW protocol version 2 is implemented.

10.2.2.11 `css600_dp` Event Status Register, EVENTSTAT

The EVENTSTAT register is used by the system to signal an event to the external debugger.

SoC-600M implements the EVENTSTAT register with top-level input dp_eventstatus, connected to an output trigger of a CoreSight *Cross-Trigger Interface* (CTI) with software acknowledge. This input signal dp_eventstatus coming from CTI trigout is inverted and synchronized in the DP before it goes to the EVENTSTAT register.

Attributes

Offset

`0x0004`

Type

Read-only

Reset

`0x0000000-`

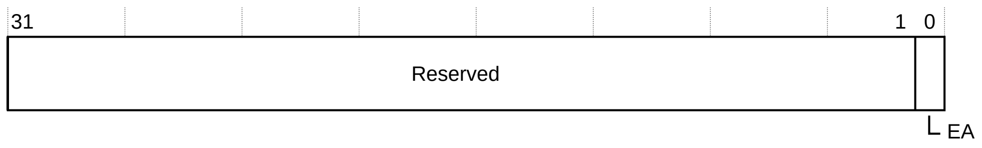
Width

32

Bit descriptions

The following figure shows the bit assignments.

Figure 10-11: EVENTSTAT register bit assignments



The following table shows the bit assignments.

Table 10-12: EVENTSTAT register bit assignments

Bits	Reset value	Name	Function
[0]	UNKNOWN	EA	Event status flag. Valid values for this bit are: **0** An event requires attention **1** There is no event requiring attention

10.2.2.12 `css600_dp` Select Register 1, SELECT1

The SELECT1 register is not used as CoreSight™ SoC-600M only supports 32-bit addressing.

Attributes

Offset

`0x0004`

Type

Write-only

Reset

`0x00000000`

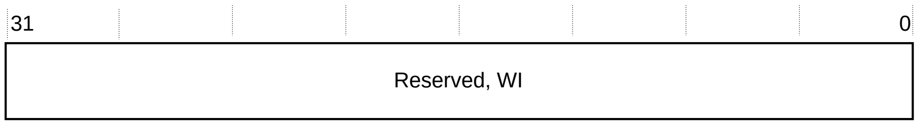
Width

32

Bit descriptions

The following figure shows the bit assignments.

Figure 10-12: SELECT1 register bit assignments



The following table shows the bit assignments.

Table 10-13: SELECT1 register bit assignments

Bits	Reset value	Name	Function
[31:0]	`0x0`	Reserved	Not used

10.2.2.13 `css600_dp` Read Resend Register, RESEND

The RESEND register enables the read data to be recovered from a corrupted debugger transfer without repeating the original AP transfer.

For JTAG-DP, this register is Reserved and any access is RES0. For SW-DP, a read to this register does not capture new data from the AP, but returns the value that was returned by the last AP read or DP RDBUFF read.

Attributes

Offset

`0x0008`

Type

Read-only

Reset

`0x00000000`

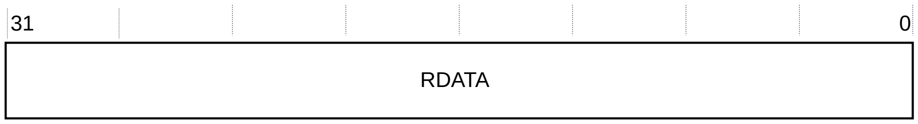
Width

32

Bit descriptions

The following figure shows the bit assignments.

Figure 10-13: RESEND register bit assignments



The following table shows the bit assignments.

Table 10-14: RESEND register bit assignments

Bits	Reset value	Name	Function
[31:0]	`0x0`	RDATA	The register can only be accessed when the DP is in SW-DP configuration. Returns last AP read or DP RDBUFF read.

10.2.2.14 `css600_dp` Select Register, SELECT

The SELECT register selects the DP address bank, and also provides the address for other components in the system, which is used by the APB Master interface on the DP to drive the APB address line.

The address on the address line driven by SELECT register is set at the start of the transfer, and does not change until the next transfer. DPIDR1.ASIZE indicates the width, in bits, of the APB master interface address bus. It is defined by the configuration parameter `APB_ADDR_WIDTH`. The DP can only issue word-aligned addresses, so paddr[1:0] are always zero. Bits [3:2] come from APACC, and higher order bits come from the SELECT register. The size of the address in SELECT is defined in DPIDR1.ASIZE. Unimplemented address bits are WI.

Attributes

Offset

`0x0080`

Type

Write-only

Reset

`0x00000000`

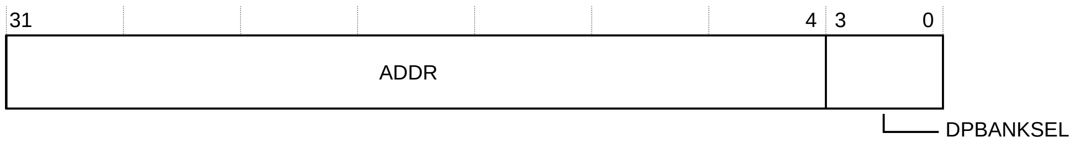
Width

32

Bit descriptions

The following figure shows the bit assignments.

Figure 10-14: SELECT register bit assignments



The following table shows the bit assignments.

Table 10-15: SELECT register bit assignments

Bits	Reset value	Name	Function
[31:4]	`0x0`	ADDR	Address Output bits [31:4]. Selects a four-word bank of system locations to access. Address bits [3:2] are provided with APACC transactions.
[3:0]	`0b0000`	DPBANKSEL	Debug Port Address bank select

10.2.2.15 `css600_dp` Read Buffer Register, RDBUFF

The RDBUFF register captures data from the AP, presented as the result of a previous read.

Access to this register is **DATA LINK DEFINED**. On JTAG-DP, Read Buffer is always RAZ/WI. On SW-DP, the behavior is as follows.

Attributes

Offset

`0x000C`

Type

Read-only

Reset

`0x00000000`

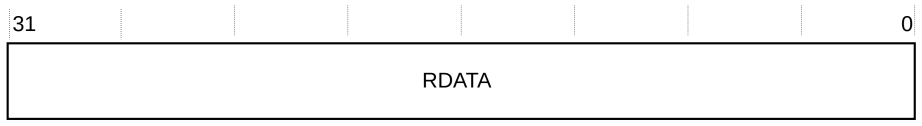
Width

32

Bit descriptions

The following figure shows the bit assignments.

Figure 10-15: RDBUFF register bit assignments



The following table shows the bit assignments.

Table 10-16: RDBUFF register bit assignments

Bits	Reset value	Name	Function
[31:0]	0x0	RDATA	Performing a read of the Read Buffer captures data from the AP, presented as the result of a previous read, without initiating a new AP transaction. This means that reading the Read Buffer returns the result of the last AP read access, without generating a new AP access. After you have read the DP Read Buffer, its contents are no longer valid. The result of a second read of the DP Read Buffer returns the result of the last AP read access.

10.2.2.16 `css600_dp` Target Selection Register, TARGETSEL

The TARGETSEL register selects the target device in a Serial Wire Debug multi-drop system. On a JTAG-DP, any access to this register is reserved, RES0. For SW-DP, the register is as shown in the description.

Attributes

Offset

`0x000C`

Type

Write-only

Reset

`0x00000000`

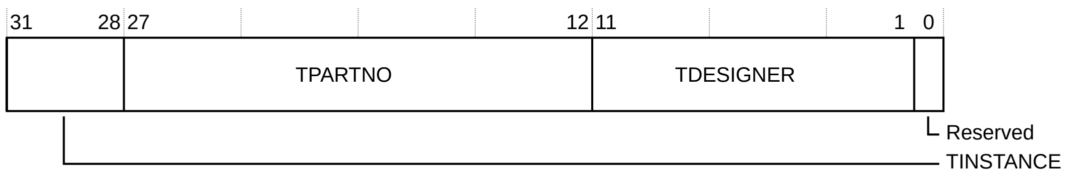
Width

32

Bit descriptions

The following figure shows the bit assignments.

Figure 10-16: TARGETSEL register bit assignments



The following table shows the bit assignments.

Table 10-17: TARGETSEL register bit assignments

Bits	Reset value	Name	Function
[31:28]	`0b0000`	TINSTANCE	SW-DP: Instance number for this device. Must be unique in a multi-drop system. Must match DLPIDR.TINSTANCE.
[27:12]	`0x0`	TPARTNO	Target part number. Must match TARGETID.TPARTNO.
[11:1]	`0b00000000000`	TDESIGNER	Designer ID. Must match TARGETID.TDESIGNER.

10.3 `css600_apbap` introduction

This section describes the programmers model of the `css600_apbap`.

10.3.1 css600_apbap register summary

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

Note

A reset value containing one or more '-' means that this register contains **UNKNOWN** or **IMPLEMENTATION-DEFINED** values. See the relevant register description for more information.

Locations that are not listed in the table are Reserved.

The 8KB memory map contains two views of the registers, one starting at `0x00000000`, and the other at `0x00001000`. In the case of RW registers, the two views provide independent physical registers. Writing to a RW register in one view does not affect the contents of the same register in the other view. For all read-only registers, the two views provide read access to the same physical register. In this case, reading from either view results in the same data being read.

Table 10-18: css600_apbap register summary

Offset	Name	Type	Reset	Width	Description
`0x0000`	DAR0	RW	`0x--------`	32	css600_apbap Direct Access Register 0, DAR0
`0x0004`	DAR1	RW	`0x--------`	32	css600_apbap Direct Access Register 1, DAR1
`0x0008`	DAR2	RW	`0x--------`	32	css600_apbap Direct Access Register 2, DAR2
...	...	...	...	...	...
`0x03FC`	DAR255	RW	`0x--------`	32	css600_apbap Direct Access Register 255, DAR255
`0x0D00`	CSW	RW	`0x30-000-2`	32	css600_apbap Control Status Word register, CSW
`0x0D04`	TAR	RW	`0x--------`	32	css600_apbap Transfer Address Register, TAR
`0x0D0C`	DRW	RW	`0x--------`	32	css600_apbap Data Read/Write register, DRW
`0x0D10`	BD0	RW	`0x--------`	32	css600_apbap Banked Data register 0, BD0
`0x0D14`	BD1	RW	`0x--------`	32	css600_apbap Banked Data register 1, BD1

Offset	Name	Type	Reset	Width	Description
0x0D18	BD2	RW	0x--------	32	css600_apbap Banked Data register 2, BD2
0x0D1C	BD3	RW	0x--------	32	css600_apbap Banked Data register 3, BD3
0x0D24	TRR	RW	0x00000000	32	css600_apbap Transfer Response Register, TRR
0x0DF4	CFG	RO	0x000101A0	32	css600_apbap Configuration register, CFG
0x0DF8	BASE	RO	0x-----00-	32	css600_apbap Debug Base Address register, BASE
0x0DFC	IDR	RO	0x34770006	32	css600_apbap Identification Register, IDR
0x0EFC	ITSTATUS	RW	0x00000000	32	css600_apbap Integration Test Status register, ITSTATUS
0x0F00	ITCTRL	RW	0x00000000	32	css600_apbap Integration Mode Control Register, ITCTRL
0x0FA0	CLAIMSET	RW	0x00000003	32	css600_apbap Claim Tag Set Register, CLAIMSET
0x0FA4	CLAIMCLR	RW	0x00000000	32	css600_apbap Claim Tag Clear Register, CLAIMCLR
0x0FB8	AUTHSTATUS	RO	0x000000--	32	css600_apbap Authentication Status Register, AUTHSTATUS
0x0FBC	DEVARCH	RO	0x47700A17	32	css600_apbap Device Architecture Register, DEVARCH
0x0FCC	DEVTYPE	RO	0x00000000	32	css600_apbap Device Type Identifier Register, DEVTYPE
0x0FD0	PIDR4	RO	0x00000004	32	css600_apbap Peripheral Identification Register 4, PIDR4
0x0FD4	PIDR5	RO	0x00000000	32	css600_apbap Peripheral Identification Register 5, PIDR5
0x0FD8	PIDR6	RO	0x00000000	32	css600_apbap Peripheral Identification Register 6, PIDR6
0x0FDC	PIDR7	RO	0x00000000	32	css600_apbap Peripheral Identification Register 7, PIDR7
0x0FE0	PIDR0	RO	0x000000E2	32	css600_apbap Peripheral Identification Register 0, PIDR0
0x0FE4	PIDR1	RO	0x000000B9	32	css600_apbap Peripheral Identification Register 1, PIDR1
0x0FE8	PIDR2	RO	0x0000003B	32	css600_apbap Peripheral Identification Register 2, PIDR2
0x0FEC	PIDR3	RO	0x00000000	32	css600_apbap Peripheral Identification Register 3, PIDR3
0x0FF0	CIDR0	RO	0x0000000D	32	css600_apbap Component Identification Register 0, CIDR0
0x0FF4	CIDR1	RO	0x00000090	32	css600_apbap Component Identification Register 1, CIDR1
0x0FF8	CIDR2	RO	0x00000005	32	css600_apbap Component Identification Register 2, CIDR2
0x0FFC	CIDR3	RO	0x000000B1	32	css600_apbap Component Identification Register 3, CIDR3
0x1000	DAR0	RW	0x--------	32	css600_apbap Direct Access Register 0, DAR0
0x1004	DAR1	RW	0x--------	32	css600_apbap Direct Access Register 1, DAR1
0x1008	DAR2	RW	0x--------	32	css600_apbap Direct Access Register 2, DAR2
...	...	...	...	...	...
0x13FC	DAR255	RW	0x--------	32	css600_apbap Direct Access Register 255, DAR255
0x1D00	CSW	RW	0x30-000-2	32	css600_apbap Control Status Word register, CSW
0x1D04	TAR	RW	0x--------	32	css600_apbap Transfer Address Register, TAR
0x1D0C	DRW	RW	0x--------	32	css600_apbap Data Read/Write register, DRW
0x1D10	BD0	RW	0x--------	32	css600_apbap Banked Data register 0, BD0
0x1D14	BD1	RW	0x--------	32	css600_apbap Banked Data register 1, BD1
0x1D18	BD2	RW	0x--------	32	css600_apbap Banked Data register 2, BD2
0x1D1C	BD3	RW	0x--------	32	css600_apbap Banked Data register 3, BD3
0x1D24	TRR	RW	0x00000000	32	css600_apbap Transfer Response Register, TRR
0x1DF4	CFG	RO	0x000101A0	32	css600_apbap Configuration register, CFG
0x1DF8	BASE	RO	0x-----00-	32	css600_apbap Debug Base Address register, BASE

Offset	Name	Type	Reset	Width	Description
`0x1DFC`	IDR	RO	`0x34770006`	32	css600_apbap Identification Register, IDR
`0x1EFC`	ITSTATUS	RW	`0x00000000`	32	css600_apbap Integration Test Status register, ITSTATUS
`0x1F00`	ITCTRL	RW	`0x00000000`	32	css600_apbap Integration Mode Control Register, ITCTRL
`0x1FA0`	CLAIMSET	RW	`0x00000003`	32	css600_apbap Claim Tag Set Register, CLAIMSET
`0x1FA4`	CLAIMCLR	RW	`0x00000000`	32	css600_apbap Claim Tag Clear Register, CLAIMCLR
`0x1FB8`	AUTHSTATUS	RO	`0x000000--`	32	css600_apbap Authentication Status Register, AUTHSTATUS
`0x1FBC`	DEVARCH	RO	`0x47700A17`	32	css600_apbap Device Architecture Register, DEVARCH
`0x1FCC`	DEVTYPE	RO	`0x00000000`	32	css600_apbap Device Type Identifier Register, DEVTYPE
`0x1FD0`	PIDR4	RO	`0x00000004`	32	css600_apbap Peripheral Identification Register 4, PIDR4
`0x1FD4`	PIDR5	RO	`0x00000000`	32	css600_apbap Peripheral Identification Register 5, PIDR5
`0x1FD8`	PIDR6	RO	`0x00000000`	32	css600_apbap Peripheral Identification Register 6, PIDR6
`0x1FDC`	PIDR7	RO	`0x00000000`	32	css600_apbap Peripheral Identification Register 7, PIDR7
`0x1FE0`	PIDR0	RO	`0x000000E2`	32	css600_apbap Peripheral Identification Register 0, PIDR0
`0x1FE4`	PIDR1	RO	`0x000000B9`	32	css600_apbap Peripheral Identification Register 1, PIDR1
`0x1FE8`	PIDR2	RO	`0x0000003B`	32	css600_apbap Peripheral Identification Register 2, PIDR2
`0x1FEC`	PIDR3	RO	`0x00000000`	32	css600_apbap Peripheral Identification Register 3, PIDR3
`0x1FF0`	CIDR0	RO	`0x0000000D`	32	css600_apbap Component Identification Register 0, CIDR0
`0x1FF4`	CIDR1	RO	`0x00000090`	32	css600_apbap Component Identification Register 1, CIDR1
`0x1FF8`	CIDR2	RO	`0x00000005`	32	css600_apbap Component Identification Register 2, CIDR2
`0x1FFC`	CIDR3	RO	`0x000000B1`	32	css600_apbap Component Identification Register 3, CIDR3

10.3.2 Register descriptions

This section describes the `css600_apbap` registers.

css600_apbap register summary provides cross references to individual registers.

10.3.2.1 css600_apbap Direct Access Register 0, DAR0

The Direct Access Registers provide a mechanism for directly mapping locations in the target memory system that is connected to the APB master interface.

Attributes

Offset

`0x0000`

Type

Read-write

Reset

`0x--------`

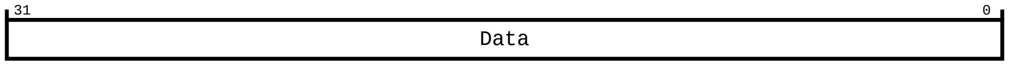
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-17: css600_apbap_DAR0



The following table shows the bit assignments.

Table 10-19: DAR0 attributes

Bits	Reset value	Name	Type	Function
[31:0]	UNKNOWN	Data	Read-write	Maps to memory address ((TAR & `0xFFFFFC00`) + `0x0`). In read mode, the register contains the data value that was read from memory, and in write mode the register contains the data value to write to memory.

10.3.2.2 css600_apbap Direct Access Register 1, DAR1

The Direct Access Registers provide a mechanism for directly mapping locations in the target memory system that is connected to the APB master interface.

Attributes

Offset

`0x0004`

Type

Read-write

Reset

`0x--------`

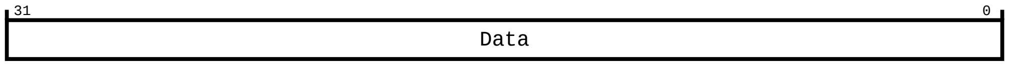
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-18: css600_apbap_DAR1



The following table shows the bit assignments.

Table 10-20: DAR1 attributes

Bits	Reset value	Name	Type	Function
[31:0]	UNKNOWN	Data	Read-write	Maps to memory address ((TAR & `0xFFFFFC00`) + `0x4`). In read mode, the register contains the data value that was read from memory, and in write mode the register contains the data value to write to memory.

10.3.2.3 css600_apbap Direct Access Register 2, DAR2

The Direct Access Registers provide a mechanism for directly mapping locations in the target memory system that is connected to the APB master interface.

Attributes

Offset

`0x0008`

Type

Read-write

Reset

`0x--------`

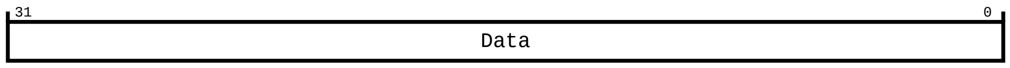
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-19: css600_apbap_DAR2



The following table shows the bit assignments.

Table 10-21: DAR2 attributes

Bits	Reset value	Name	Type	Function
[31:0]	UNKNOWN	Data	Read-write	Maps to memory address ((TAR & `0xFFFFFC00`) + `0x8`). In read mode, the register contains the data value that was read from memory, and in write mode the register contains the data value to write to memory.

10.3.2.4 css600_apbap Direct Access Register 255, DAR255

The Direct Access Registers provide a mechanism for directly mapping locations in the target memory system that is connected to the APB master interface.

Attributes

Offset

`0x03FC`

Type

Read-write

Reset

`0x--------`

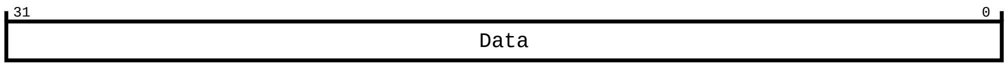
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-20: css600_apbap_DAR255



The following table shows the bit assignments.

Table 10-22: DAR255 attributes

Bits	Reset value	Name	Type	Function
[31:0]	UNKNOWN	Data	Read-write	Maps to memory address ((TAR & `0xFFFFFC00`) + `0x3FC`). In read mode, the register contains the data value that was read from memory, and in write mode the register contains the data value to write to memory.

10.3.2.5 css600_apbap Control Status Word register, CSW

The CSW register configures and controls accesses through the APB master interface to the connected memory system.

Attributes

Offset

`0x0D00`

Type

Read-write

Reset

`0x30-000-2`

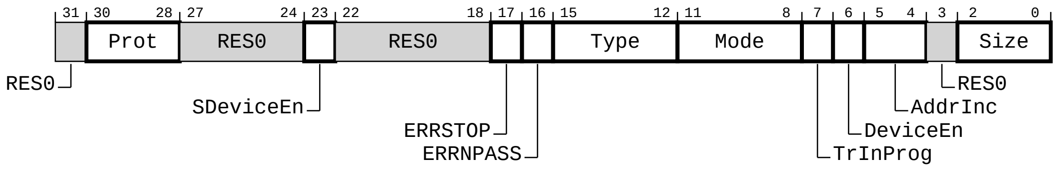
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-21: css600_apbap_CSW



The following table shows the bit assignments.

Table 10-23: CSW attributes

Bits	Reset value	Name	Type	Function
[31]	`0b0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[30:28]	`0b011`	Prot	Read-write	Drives APB master interface pprot_m[2:0] which specifies the APB4 protection encoding. The reset value is `0x3` (Data, Non-secure, Privileged). Together with the Access Port Enable interface signals, CSW.Prot[1] determines whether a secure access is allowed on the master interface. Accesses are permitted as follows: access_permitted = (ap_en && ap_secure_en) || (ap_en && CSW.Prot[1]).
[27:24]	`0b0000`	RES0	Read-write	Reserved bit or field with SBZP behavior
[23]	UNKNOWN	SDeviceEn	Read-only	Indicates the status of the ap_en and ap_secure_en ports. It is set when both ap_en and ap_secure_en are HIGH, but otherwise is clear. If this bit is clear, Secure APB transfers are not permitted. Non-secure memory accesses and internal register accesses that do not initiate memory accesses are permitted regardless of the status of this bit.
[22:18]	`0b00000`	RES0	Read-write	Reserved bit or field with SBZP behavior
[17]	`0b0`	ERRSTOP	Read-write	Stop on error. Reset to 0. **0** Memory access errors do not prevent future memory accesses **1** Memory access errors prevent future memory accesses
[16]	`0b0`	ERRNPASS	Read-write	Errors are not passed upstream: **0** Memory access errors are passed upstream **1** Memory access errors are not passed upstream
[15:12]	`0b0000`	Type	Read-only	This field is reserved. Reads return `0x0` and writes are ignored.

Bits	Reset value	Name	Type	Function
[11:8]	`0b0000`	Mode	Read-only	Specifies the mode of operation. Reset to `0x0`. All other values are reserved. **`0x0`** Normal download or upload mode
[7]	`0b0`	TrInProg	Read-only	Transfer in progress. This field indicates whether a transfer is in progress on the APB master interface.
[6]	UNKNOWN	DeviceEn	Read-only	Indicates the status of the ap_en port. The bit is set when ap_en is HIGH, but otherwise is clear. If this bit is clear, no APB transfers are carried out, that is, both Secure and Non-secure accesses are blocked.
[5:4]	`0b00`	AddrInc	Read-write	Auto address increment mode on RW data access. Only increments if the current transaction completes without an error response and the transaction is not aborted. Reset to `0b0`. **`0x0`** Auto increment OFF **`0x1`** Increment, single. Single transfer from corresponding byte lane. **`0x2`** Reserved **`0x3`** Reserved
[3]	`0b0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[2:0]	`0b010`	Size	Read-only	Size of the data access to perform. The APB-AP supports only word accesses and this field is fixed at `0x2`. The reset value is `0x2`.

10.3.2.6 css600_apbap Transfer Address Register, TAR

TAR holds the transfer address of the current transfer. TAR must be programmed before initiating any memory transfer through DRW, or Banked Data Registers, or Direct Access Registers.

Attributes

Offset

`0x0D04`

Type

Read-write

Reset

`0x--------`

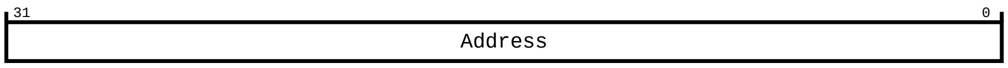
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-22: css600_apbap_TAR



The following table shows the bit assignments.

Table 10-24: TAR attributes

Bits	Reset value	Name	Type	Function
[31:0]	UNKNOWN	Address	Read-write	Address of the current transfer. When a memory access is initiated by accessing the DRW register, the TAR value directly gives the 32-bit transfer address. When a memory access is initiated by accessing Banked Data registers, the TAR only provides the upper bits [31:4] and the remaining address bits [3:0] come from the offset of Banked Data register being accessed. When a memory access is initiated by accessing Direct Access Registers, the TAR provides the upper bits [31:10] and the remaining address bits [9:0] come from the offset of the DAR being accessed.

10.3.2.7 css600_apbap Data Read/Write register, DRW

A write to the DRW register initiates a memory write transaction on the master. AP drives DRW write data on the data bus during the data phase of the current transfer. Reading the DRW register initiates a memory read transaction on the master. The resulting read data that is received from the memory system is returned on the slave interface.

Attributes

Offset

`0x0D0C`

Type

Read-write

Reset

`0x--------`

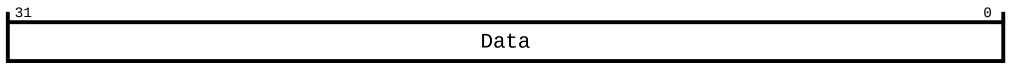
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-23: css600_apbap_DRW



The following table shows the bit assignments.

Table 10-25: DRW attributes

Bits	Reset value	Name	Type	Function
[31:0]	UNKNOWN	Data	Read-write	Current transfer data value. In read mode, the register contains the data value that was read from the current transfer, and in write mode the register contains the data value to write for the current transfer.

10.3.2.8 css600_apbap Banked Data register 0, BD0

The Banked Data registers provide a mechanism for directly mapping APB slave accesses to memory transfers without having to rewrite the TAR within a 16-byte boundary.

Attributes

Offset

`0x0D10`

Type

Read-write

Reset

`0x--------`

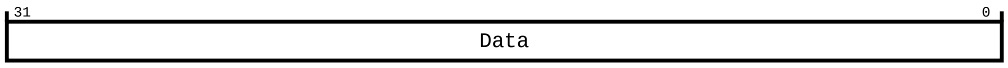
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-24: css600_apbap_BD0



The following table shows the bit assignments.

Table 10-26: BD0 attributes

Bits	Reset value	Name	Type	Function
[31:0]	UNKNOWN	Data	Read-write	Maps to memory address ((TAR & `0xFFFFFFF0`) + `0x0`). In read mode, the register contains the data value that was read from the current transfer, and in write mode the register contains the data value to write for the current transfer.

10.3.2.9 css600_apbap Banked Data register 1, BD1

The Banked Data registers provide a mechanism for directly mapping APB slave accesses to memory transfers without having to rewrite the TAR within a 16-byte boundary.

Attributes

Offset

`0x0D14`

Type

Read-write

Reset

`0x--------`

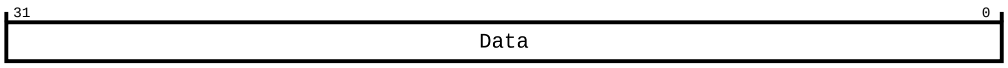
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-25: css600_apbap_BD1



The following table shows the bit assignments.

Table 10-27: BD1 attributes

Bits	Reset value	Name	Type	Function
[31:0]	UNKNOWN	Data	Read-write	Maps to memory address ((TAR & `0xFFFFFFF0`) + `0x4`). In read mode, the register contains the data value that was read from the current transfer, and in write mode the register contains the data value to write for the current transfer.

10.3.2.10 css600_apbap Banked Data register 2, BD2

The Banked Data registers provide a mechanism for directly mapping APB slave accesses to memory transfers without having to rewrite the TAR within a 16-byte boundary.

Attributes

Offset

`0x0D18`

Type

Read-write

Reset

`0x--------`

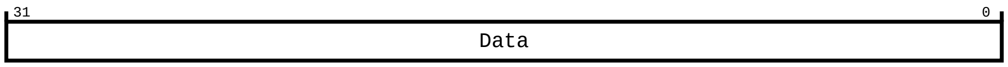
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-26: css600_apbap_BD2



The following table shows the bit assignments.

Table 10-28: BD2 attributes

Bits	Reset value	Name	Type	Function
[31:0]	UNKNOWN	Data	Read-write	Maps to memory address ((TAR & `0xFFFFFFF0`) + `0x8`). In read mode, the register contains the data value that was read from the current transfer, and in write mode the register contains the data value to write for the current transfer.

10.3.2.11 css600_apbap Banked Data register 3, BD3

The Banked Data registers provide a mechanism for directly mapping APB slave accesses to memory transfers without having to rewrite the TAR within a 16-byte boundary.

Attributes

Offset

`0x0D1C`

Type

Read-write

Reset

`0x--------`

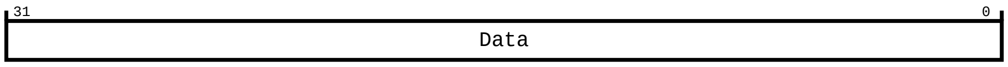
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-27: css600_apbap_BD3



The following table shows the bit assignments.

Table 10-29: BD3 attributes

Bits	Reset value	Name	Type	Function
[31:0]	UNKNOWN	Data	Read-write	Maps to memory address ((TAR & `0xFFFFFFF0`) + `0xC`). In read mode, the register contains the data value that was read from the current transfer, and in write mode the register contains the data value to write for the current transfer.

10.3.2.12 css600_apbap Transfer Response Register, TRR

The Transfer Response Register is used to capture an error response received during a transaction. It is also used to clear any logged responses.

Attributes

Offset

`0x0D24`

Type

Read-write

Reset

`0x00000000`

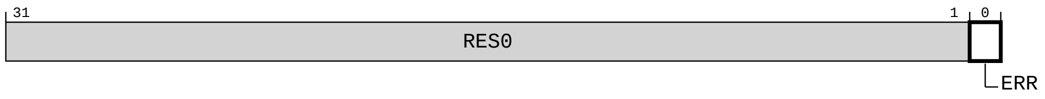
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-28: css600_apbap_TRR



The following table shows the bit assignments.

Table 10-30: TRR attributes

Bits	Reset value	Name	Type	Function
[31:1]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[0]	`0b0`	ERR	Read-write	Logged error. **0** On reads - no error response logged. Writing to this bit has no effect. **1** On reads - error response logged. Writing to this bit clears this bit to 0.

10.3.2.13 css600_apbap Configuration register, CFG

The CFG register is the css_600 configuration register.

Attributes

Offset

`0x0DF4`

Type

Read-only

Reset

`0x000101A0`

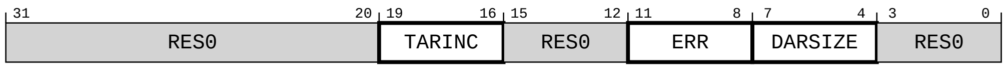
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-29: css600_apbap_CFG



The following table shows the bit assignments.

Table 10-31: CFG attributes

Bits	Reset value	Name	Type	Function
[31:20]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[19:16]	`0b0001`	TARINC	Read-only	TAR incrementer size. Returns `0x1` indicating a TAR incrementer size of 10-bits.
[15:12]	`0b0000`	RES0	Read-only	Reserved bit or field with SBZP behavior
[11:8]	`0b0001`	ERR	Read-only	Indicates the type of error handling that is implemented: **`0x0`** Error response handling 0. CSW.ERRNPASS, CSW.ERRSTOP, and TRR are not implemented. **`0x1`** Error response handling 1. CSW.ERRNPASS, CSW.ERRSTOP, and TRR are implemented.
[7:4]	`0b1010`	DARSIZE	Read-only	Size of DAR register space. Returns `0xA` indicating that 1KB (256 registers, each 32-bit wide) of DAR is implemented.
[3:0]	`0b0000`	RES0	Read-only	Reserved bit or field with SBZP behavior

10.3.2.14 css600_apbap Debug Base Address register, BASE

Provides an initial system address for the first component in the system. Typically, the system address is the address of a top-level ROM Table that indicates where APv2 APs are located.

Attributes

Offset

`0x0DF8`

Type

Read-only

Reset

`0x-----00-`

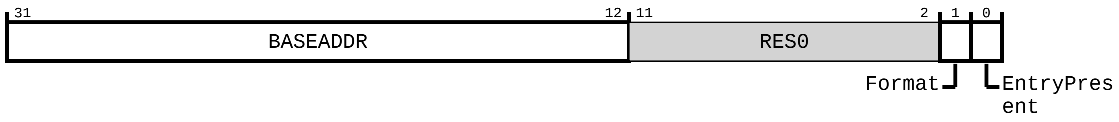
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-30: css600_apbap_BASE



The following table shows the bit assignments.

Table 10-32: BASE attributes

Bits	Reset value	Name	Type	Function
[31:12]	**IMPLEMENTATION DEFINED**	BASEADDR	Read-only	Base address of a ROM table. It points to the start of the debug register space or a ROM table address. Bits[11:0] of the address are `0x000` because the address is aligned to 4KB boundary. This field is valid only if BASE.EntryPresent bit is set to 1, in which case it returns the tie-off value of the input signal baseaddr[31:12], otherwise, it reads as `0x0`.
[11:2]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[1]	`0b1`	Format	Read-only	Base address register format. Returns the value `0b1` indicating the ADIv5 format, which is unchanged in ADIv6.
[0]	**IMPLEMENTATION DEFINED**	EntryPresent	Read-only	This field indicates whether a debug component is present for this AP. It returns the tie-off value of the input signal baseaddr_valid. **0** No debug entry present. **1** Debug entry present and BASE.BASEADDR indicate the start address of the debug register space or ROM table.

10.3.2.15 css600_apbap Identification Register, IDR

This register provides a mechanism for the debugger to know various identity attributes of the AP.

Attributes

Offset

`0x0DFC`

Type

Read-only

Reset

`0x34770006`

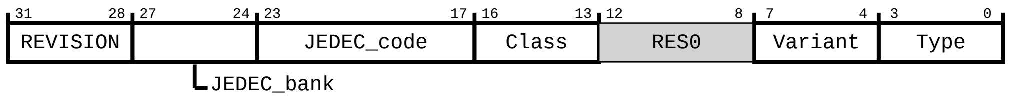
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-31: css600_apbap_IDR



The following table shows the bit assignments.

Table 10-33: IDR attributes

Bits	Reset value	Name	Type	Function
[31:28]	`0b0011`	REVISION	Read-only	Revision. An incremental value starting at `0x0` for the first design of a component. See the Component list in Chapter 1 for information on the RTL revision of the component.
[27:24]	`0b0100`	JEDEC_bank	Read-only	The JEP106 continuation code. Returns `0x4`, indicating Arm as the designer.
[23:17]	`0x3B`	JEDEC_code	Read-only	The JEP106 identification code. Returns `0x3B`, indicating Arm as the designer.
[16:13]	`0b1000`	Class	Read-only	Returns `0x8`, indicating that this is a Memory Access Port
[12:8]	`0b00000`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0000`	Variant	Read-only	Returns `0x0`, indicating no variation from base type specified by IDR.Type
[3:0]	`0b0110`	Type	Read-only	Returns `0x6`, indicating that this is an APB4 Access Port

10.3.2.16 css600_apbap Integration Test Status register, ITSTATUS

This register indicates the Integration Test DP Abort status.

Attributes

Offset

`0x0EFC`

Type

Read-write

Reset

`0x00000000`

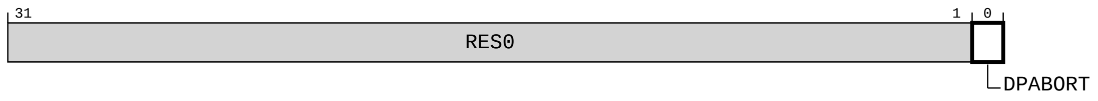
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-32: css600_apbap_ITSTATUS



The following table shows the bit assignments.

Table 10-34: ITSTATUS attributes

Bits	Reset value	Name	Type	Function
[31:1]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[0]	`0b0`	DPABORT	Read-only	When in Integration testing mode (ITCTRL.IME=`0b1`): Behaves as a sticky bit and latches to 1 on a rising edge of dp_abort. Cleared on a read from this register. If dp_abort rises in the same cycle as a read of the ITSTATUS register is received, the read takes priority and the register is cleared. When in normal functional operation mode (ITCTRL.IME=`0b0`): Read as 0, writes ignored.

10.3.2.17 css600_apbap Integration Mode Control Register, ITCTRL

The Integration Mode Control register is used to enable topology detection.

Attributes

Offset

`0x0F00`

Type

Read-write

Reset

`0x00000000`

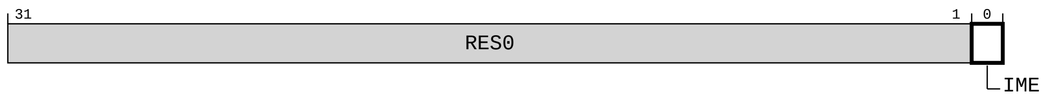
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-33: css600_apbap_ITCTRL



The following table shows the bit assignments.

Table 10-35: ITCTRL attributes

Bits	Reset value	Name	Type	Function
[31:1]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[0]	`0b0`	IME	Read-write	Integration Mode Enable. When set, the component enters integration mode, enabling topology detection or integration testing to be performed.

10.3.2.18 css600_apbap Claim Tag Set Register, CLAIMSET

This register forms one half of the claim tag value. On writes, this location enables individual bits to be set. On reads, it returns the number of bits that can be set.

Attributes

Offset

`0x0FA0`

Type

Read-write

Reset

`0x00000003`

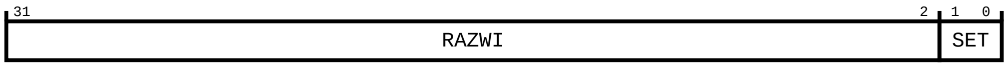
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-34: css600_apbap_CLAIMSET



The following table shows the bit assignments.

Table 10-36: CLAIMSET attributes

Bits	Reset value	Name	Type	Function
[31:2]	`0x0`	RAZ/ WI	Read-write	RAZ/WI
[1:0]	`0b11`	SET	Read-write	A bit-programmable register bank that sets the claim tag value. A read returns a logic 1 for all implemented locations.

10.3.2.19 css600_apbap Claim Tag Clear Register, CLAIMCLR

This register forms one half of the claim tag value. On writes, this location enables individual bits to be cleared. On reads, it returns the current claim tag value.

Attributes

Offset

`0x0FA4`

Type

Read-write

Reset

`0x00000000`

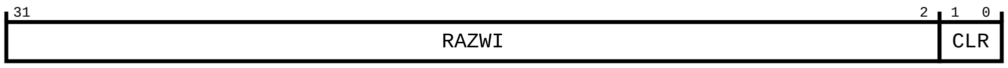
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-35: css600_apbap_CLAIMCLR



The following table shows the bit assignments.

Table 10-37: CLAIMCLR attributes

Bits	Reset value	Name	Type	Function
[31:2]	`0x0`	RAZ/ WI	Read-write	RAZ/WI
[1:0]	`0b00`	CLR	Read-write	A bit-programmable register bank that clears the claim tag value. It is zero at reset. It is used by software agents to signal to each other ownership of the hardware. It has no direct effect on the hardware itself.

10.3.2.20 css600_apbap Authentication Status Register, AUTHSTATUS

Reports the current status of the authentication control signals.

Attributes

Offset

`0x0FB8`

Type

Read-only

Reset

`0x000000--`

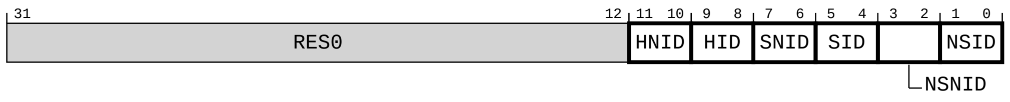
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-36: css600_apbap_AUTHSTATUS



The following table shows the bit assignments.

Table 10-38: AUTHSTATUS attributes

Bits	Reset value	Name	Type	Function
[31:12]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[11:10]	`0b00`	HNID	Read-only	Hypervisor non-invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.

Bits	Reset value	Name	Type	Function
[9:8]	`0b00`	HID	Read-only	Hypervisor invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.
[7:6]	UNKNOWN	SNID	Read-only	Secure non-invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.
[5:4]	UNKNOWN	SID	Read-only	Secure invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.
[3:2]	UNKNOWN	NSNID	Read-only	Non-secure non-invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.
[1:0]	UNKNOWN	NSID	Read-only	Non-secure invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.

10.3.2.21 css600_apbap Device Architecture Register, DEVARCH

Identifies the architect and architecture of a CoreSight component. The architect might differ from the designer of a component, for example Arm defines the architecture but another company designs and implements the component.

Attributes

Offset

`0x0FBC`

Type

Read-only

Reset

`0x47700A17`

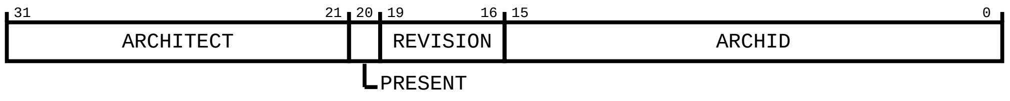
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-37: css600_apbap_DEVARCH



The following table shows the bit assignments.

Table 10-39: DEVARCH attributes

Bits	Reset value	Name	Type	Function
[31:21]	`0x23B`	ARCHITECT	Read-only	Returns 0x23B, denoting Arm as architect of the component.
[20]	`0b1`	PRESENT	Read-only	Returns 1, indicating that the DEVARCH register is present.
[19:16]	`0b0000`	REVISION	Read-only	Architecture revision. Returns the revision of the architecture that the ARCHID field specifies.
[15:0]	`0xA17`	ARCHID	Read-only	Architecture ID. Returns 0x0A17, identifying APv2 MEM-AP architecture v0.

10.3.2.22 css600_apbap Device Type Identifier Register, DEVTYPE

A debugger can use this register to get information about a component that has an unrecognized Part number.

Attributes

Offset

`0x0FCC`

Type

Read-only

Reset

`0x00000000`

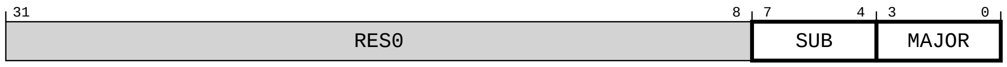
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-38: css600_apbap_DEVTYPE



The following table shows the bit assignments.

Table 10-40: DEVTYPE attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0000`	SUB	Read-only	Minor classification. Returns `0x0`, Other/undefined.
[3:0]	`0b0000`	MAJOR	Read-only	Major classification. Returns `0x0`, Miscellaneous.

10.3.2.23 css600_apbap Peripheral Identification Register 4, PIDR4

The PIDR4 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FD0`

Type

Read-only

Reset

`0x00000004`

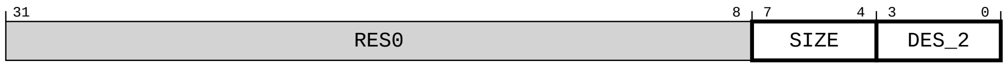
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-39: css600_apbap_PIDR4



The following table shows the bit assignments.

Table 10-41: PIDR4 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior

Bits	Reset value	Name	Type	Function
[7:4]	`0b0000`	SIZE	Read-only	Indicates the memory size that is used by this component. Returns 0 indicating that the component uses an **UNKNOWN** number of 4KB blocks. Using the SIZE field to indicate the size of the component is deprecated.
[3:0]	`0b0100`	DES_2	Read-only	JEP106 continuation code. Together, with PIDR2.DES_1 and PIDR1.DES_0, they indicate the designer of the component and not the implementer, except where the two are the same.

10.3.2.24 css600_apbap Peripheral Identification Register 5, PIDR5

The PIDR5 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FD4`

Type

Read-only

Reset

`0x00000000`

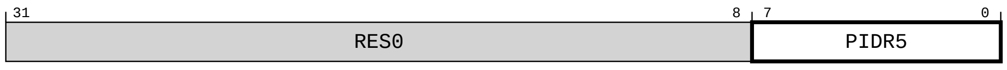
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-40: css600_apbap_PIDR5



The following table shows the bit assignments.

Table 10-42: PIDR5 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	PIDR5	Read-only	Reserved.

10.3.2.25 css600_apbap Peripheral Identification Register 6, PIDR6

The PIDR6 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FD8`

Type

Read-only

Reset

`0x00000000`

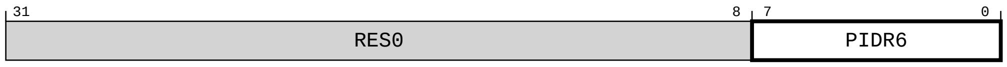
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-41: css600_apbap_PIDR6



The following table shows the bit assignments.

Table 10-43: PIDR6 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	PIDR6	Read-only	Reserved.

10.3.2.26 css600_apbap Peripheral Identification Register 7, PIDR7

The PIDR7 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FDC`

Type

Read-only

Reset

`0x00000000`

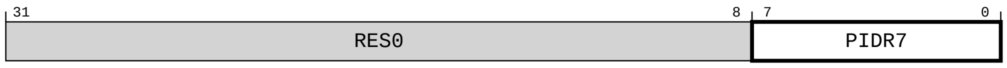
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-42: css600_apbap_PIDR7



The following table shows the bit assignments.

Table 10-44: PIDR7 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	PIDR7	Read-only	Reserved.

10.3.2.27 css600_apbap Peripheral Identification Register 0, PIDR0

The PIDR0 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FE0`

Type

Read-only

Reset

`0x000000E2`

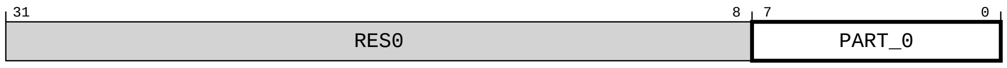
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-43: css600_apbap_PIDR0



The following table shows the bit assignments.

Table 10-45: PIDR0 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	0xE2	PART_0	Read-only	Part number, bits[7:0]. Taken together with PIDR1.PART_1 it indicates the component. The Part Number is selected by the designer of the component.

10.3.2.28 css600_apbap Peripheral Identification Register 1, PIDR1

The PIDR1 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FE4`

Type

Read-only

Reset

`0x000000B9`

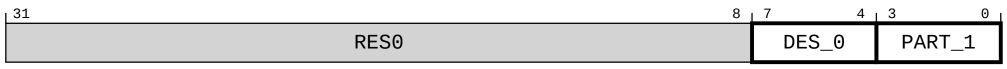
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-44: css600_apbap_PIDR1



The following table shows the bit assignments.

Table 10-46: PIDR1 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b1011`	DES_0	Read-only	JEP106 identification code, bits[3:0]. Together, with PIDR4.DES_2 and PIDR2.DES_1, they indicate the designer of the component and not the implementer, except where the two are the same.
[3:0]	`0b1001`	PART_1	Read-only	Part number, bits[11:8]. Taken together with PIDR0.PART_0 it indicates the component. The Part Number is selected by the designer of the component.

10.3.2.29 css600_apbap Peripheral Identification Register 2, PIDR2

The PIDR2 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FE8`

Type

Read-only

Reset

`0x0000003B`

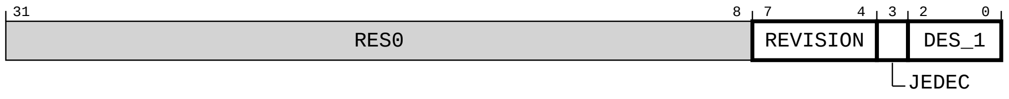
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-45: css600_apbap_PIDR2



The following table shows the bit assignments.

Table 10-47: PIDR2 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0011`	REVISION	Read-only	Revision. It is an incremental value starting at `0x0` for the first design of a component. See the Component list in Chapter 1 for information on the RTL revision of the component.
[3]	`0b1`	JEDEC	Read-only	1 - Always set. Indicates that a JEDEC assigned value is used.
[2:0]	`0b011`	DES_1	Read-only	JEP106 identification code, bits[6:4]. Together, with PIDR4.DES_2 and PIDR1.DES_0, they indicate the designer of the component and not the implementer, except where the two are the same.

10.3.2.30 css600_apbap Peripheral Identification Register 3, PIDR3

The PIDR3 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FEC`

Type

Read-only

Reset

`0x00000000`

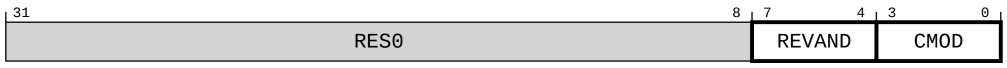
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-46: css600_apbap_PIDR3



The following table shows the bit assignments.

Table 10-48: PIDR3 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0000`	REVAND	Read-only	This field indicates minor errata fixes specific to this design, for example metal fixes after implementation. In most cases this field is `0x0`.
[3:0]	`0b0000`	CMOD	Read-only	Customer Modified. Where the component is reusable IP, this value indicates if the customer has modified the behavior of the component. In most cases this field is `0x0`.

10.3.2.31 css600_apbap Component Identification Register 0, CIDR0

The CIDR0 register is part of the set of component identification registers.

Attributes

Offset

`0x0FF0`

Type

Read-only

Reset

`0x0000000D`

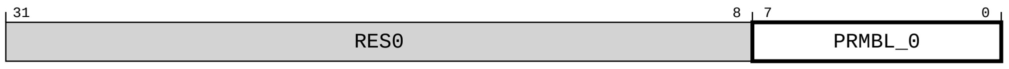
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-47: css600_apbap_CIDR0



The following table shows the bit assignments.

Table 10-49: CIDR0 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0xD`	PRMBL_0	Read-only	Preamble. Returns `0x0D`.

10.3.2.32 css600_apbap Component Identification Register 1, CIDR1

The CIDR1 register is part of the set of component identification registers.

Attributes

Offset

`0x0FF4`

Type

Read-only

Reset

`0x00000090`

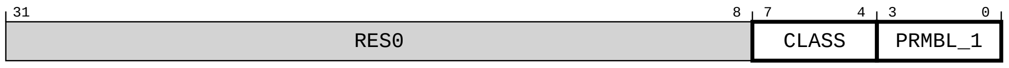
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-48: css600_apbap_CIDR1



The following table shows the bit assignments.

Table 10-50: CIDR1 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b1001`	CLASS	Read-only	Component class. Returns `0x9`, indicating this is a CoreSight component.
[3:0]	`0b0000`	PRMBL_1	Read-only	Preamble. Returns `0x0`.

10.3.2.33 css600_apbap Component Identification Register 2, CIDR2

The CIDR2 register is part of the set of component identification registers.

Attributes

Offset

`0x0FF8`

Type

Read-only

Reset

`0x00000005`

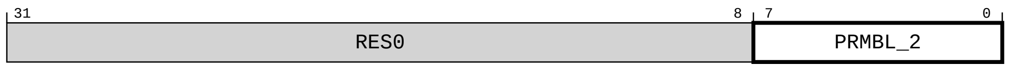
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-49: css600_apbap_CIDR2



The following table shows the bit assignments.

Table 10-51: CIDR2 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x5`	PRMBL_2	Read-only	Preamble. Returns `0x05`.

10.3.2.34 css600_apbap Component Identification Register 3, CIDR3

The CIDR3 register is part of the set of component identification registers.

Attributes

Offset

`0x0FFC`

Type

Read-only

Reset

`0x000000B1`

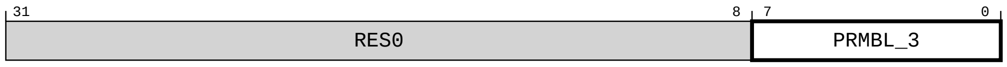
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-50: css600_apbap_CIDR3



The following table shows the bit assignments.

Table 10-52: CIDR3 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0xB1`	PRMBL_3	Read-only	Preamble. Returns `0xB1`.

10.4 `css600_ahbap` introduction

This section describes the programmers model of the `css600_ahbap`.

10.4.1 css600_ahbap register summary

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

Note

A reset value containing one or more '-' means that this register contains **UNKNOWN** or **IMPLEMENTATION-DEFINED** values. See the relevant register description for more information.

Locations that are not listed in the table are Reserved.

The 8KB memory map contains two views of the registers, one starting at `0x00000000`, and the other at `0x00001000`. In the case of RW registers, the two views provide independent physical registers. Writing to a RW register in one view does not affect the contents of the same register in the other view. For all read-only registers, the two views provide read access to the same physical register. In this case, reading from either view results in the same data being read.

Table 10-53: css600_ahbap register summary

Offset	Name	Type	Reset	Width	Description
0x0000	DAR0	RW	0x--------	32	css600_ahbap Direct Access Register 0, DAR0
0x0004	DAR1	RW	0x--------	32	css600_ahbap Direct Access Register 1, DAR1
0x0008	DAR2	RW	0x--------	32	css600_ahbap Direct Access Register 2, DAR2
…	…	…	…	…	…
0x03FC	DAR255	RW	0x--------	32	css600_ahbap Direct Access Register 255, DAR255
0x0D00	CSW	RW	0x43-000-2	32	css600_ahbap Control Status Word register, CSW
0x0D04	TAR	RW	0x--------	32	css600_ahbap Transfer Address Register, TAR
0x0D0C	DRW	RW	0x--------	32	css600_ahbap Data Read/Write register, DRW
0x0D10	BD0	RW	0x--------	32	css600_ahbap Banked Data register 0, BD0
0x0D14	BD1	RW	0x--------	32	css600_ahbap Banked Data register 1, BD1
0x0D18	BD2	RW	0x--------	32	css600_ahbap Banked Data register 2, BD2
0x0D1C	BD3	RW	0x--------	32	css600_ahbap Banked Data register 3, BD3
0x0D24	TRR	RW	0x00000000	32	css600_ahbap Transfer Response Register, TRR
0x0DF4	CFG	RO	0x000101A0	32	css600_ahbap Configuration register, CFG
0x0DF8	BASE	RO	0x-----00-	32	css600_ahbap Debug Base Address register, BASE
0x0DFC	IDR	RO	0x44770008	32	css600_ahbap Identification Register, IDR
0x0EFC	ITSTATUS	RW	0x00000000	32	css600_ahbap Integration Test Status register, ITSTATUS
0x0F00	ITCTRL	RW	0x00000000	32	css600_ahbap Integration Mode Control Register, ITCTRL
0x0FA0	CLAIMSET	RW	0x00000003	32	css600_ahbap Claim Tag Set Register, CLAIMSET
0x0FA4	CLAIMCLR	RW	0x00000000	32	css600_ahbap Claim Tag Clear Register, CLAIMCLR
0x0FB8	AUTHSTATUS	RO	0x000000--	32	css600_ahbap Authentication Status Register, AUTHSTATUS
0x0FBC	DEVARCH	RO	0x47700A17	32	css600_ahbap Device Architecture Register, DEVARCH
0x0FCC	DEVTYPE	RO	0x00000000	32	css600_ahbap Device Type Identifier Register, DEVTYPE
0x0FD0	PIDR4	RO	0x00000004	32	css600_ahbap Peripheral Identification Register 4, PIDR4
0x0FD4	PIDR5	RO	0x00000000	32	css600_ahbap Peripheral Identification Register 5, PIDR5
0x0FD8	PIDR6	RO	0x00000000	32	css600_ahbap Peripheral Identification Register 6, PIDR6
0x0FDC	PIDR7	RO	0x00000000	32	css600_ahbap Peripheral Identification Register 7, PIDR7
0x0FE0	PIDR0	RO	0x000000E3	32	css600_ahbap Peripheral Identification Register 0, PIDR0
0x0FE4	PIDR1	RO	0x000000B9	32	css600_ahbap Peripheral Identification Register 1, PIDR1
0x0FE8	PIDR2	RO	0x0000004B	32	css600_ahbap Peripheral Identification Register 2, PIDR2
0x0FEC	PIDR3	RO	0x00000000	32	css600_ahbap Peripheral Identification Register 3, PIDR3

Offset	Name	Type	Reset	Width	Description
0x0FF0	CIDR0	RO	0x0000000D	32	css600_ahbap Component Identification Register 0, CIDR0
0x0FF4	CIDR1	RO	0x00000090	32	css600_ahbap Component Identification Register 1, CIDR1
0x0FF8	CIDR2	RO	0x00000005	32	css600_ahbap Component Identification Register 2, CIDR2
0x0FFC	CIDR3	RO	0x000000B1	32	css600_ahbap Component Identification Register 3, CIDR3
0x1000	DAR0	RW	0x--------	32	css600_ahbap Direct Access Register 0, DAR0
0x1004	DAR1	RW	0x--------	32	css600_ahbap Direct Access Register 1, DAR1
0x1008	DAR2	RW	0x--------	32	css600_ahbap Direct Access Register 2, DAR2
...	...	...	...	...	...
0x13FC	DAR255	RW	0x--------	32	css600_ahbap Direct Access Register 255, DAR255
0x1D00	CSW	RW	0x43-000-2	32	css600_ahbap Control Status Word register, CSW
0x1D04	TAR	RW	0x--------	32	css600_ahbap Transfer Address Register, TAR
0x1D0C	DRW	RW	0x--------	32	css600_ahbap Data Read/Write register, DRW
0x1D10	BD0	RW	0x--------	32	css600_ahbap Banked Data register 0, BD0
0x1D14	BD1	RW	0x--------	32	css600_ahbap Banked Data register 1, BD1
0x1D18	BD2	RW	0x--------	32	css600_ahbap Banked Data register 2, BD2
0x1D1C	BD3	RW	0x--------	32	css600_ahbap Banked Data register 3, BD3
0x1D24	TRR	RW	0x00000000	32	css600_ahbap Transfer Response Register, TRR
0x1DF4	CFG	RO	0x000101A0	32	css600_ahbap Configuration register, CFG
0x1DF8	BASE	RO	0x-----00-	32	css600_ahbap Debug Base Address register, BASE
0x1DFC	IDR	RO	0x44770008	32	css600_ahbap Identification Register, IDR
0x1EFC	ITSTATUS	RW	0x00000000	32	css600_ahbap Integration Test Status register, ITSTATUS
0x1F00	ITCTRL	RW	0x00000000	32	css600_ahbap Integration Mode Control Register, ITCTRL
0x1FA0	CLAIMSET	RW	0x00000003	32	css600_ahbap Claim Tag Set Register, CLAIMSET
0x1FA4	CLAIMCLR	RW	0x00000000	32	css600_ahbap Claim Tag Clear Register, CLAIMCLR
0x1FB8	AUTHSTATUS	RO	0x000000--	32	css600_ahbap Authentication Status Register, AUTHSTATUS
0x1FBC	DEVARCH	RO	0x47700A17	32	css600_ahbap Device Architecture Register, DEVARCH
0x1FCC	DEVTYPE	RO	0x00000000	32	css600_ahbap Device Type Identifier Register, DEVTYPE
0x1FD0	PIDR4	RO	0x00000004	32	css600_ahbap Peripheral Identification Register 4, PIDR4
0x1FD4	PIDR5	RO	0x00000000	32	css600_ahbap Peripheral Identification Register 5, PIDR5
0x1FD8	PIDR6	RO	0x00000000	32	css600_ahbap Peripheral Identification Register 6, PIDR6
0x1FDC	PIDR7	RO	0x00000000	32	css600_ahbap Peripheral Identification Register 7, PIDR7
0x1FE0	PIDR0	RO	0x000000E3	32	css600_ahbap Peripheral Identification Register 0, PIDR0
0x1FE4	PIDR1	RO	0x000000B9	32	css600_ahbap Peripheral Identification Register 1, PIDR1
0x1FE8	PIDR2	RO	0x0000004B	32	css600_ahbap Peripheral Identification Register 2, PIDR2
0x1FEC	PIDR3	RO	0x00000000	32	css600_ahbap Peripheral Identification Register 3, PIDR3
0x1FF0	CIDR0	RO	0x0000000D	32	css600_ahbap Component Identification Register 0, CIDR0
0x1FF4	CIDR1	RO	0x00000090	32	css600_ahbap Component Identification Register 1, CIDR1
0x1FF8	CIDR2	RO	0x00000005	32	css600_ahbap Component Identification Register 2, CIDR2
0x1FFC	CIDR3	RO	0x000000B1	32	css600_ahbap Component Identification Register 3, CIDR3

10.4.2 Register descriptions

This section describes the `css600_ahbap` registers.

css600_ahbap register summary provides cross references to individual registers.

10.4.2.1 css600_ahbap Direct Access Register 0, DAR0

The Direct Access Registers provide a mechanism for directly mapping locations in the target memory system that is connected to the APB master interface.

Attributes

Offset

`0x0000`

Type

Read-write

Reset

`0x--------`

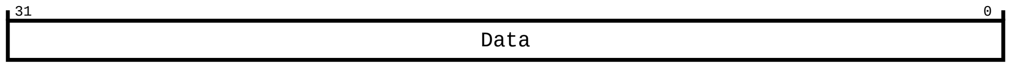
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-51: css600_ahbap_DAR0



The following table shows the bit assignments.

Table 10-54: DAR0 attributes

Bits	Reset value	Name	Type	Function
[31:0]	UNKNOWN	Data	Read-write	Maps to memory address ((TAR & `0xFFFFFC00`) + `0x0`). In read mode, the register contains the data value that was read from memory, and in write mode the register contains the data value to write to memory.

10.4.2.2 css600_ahbap Direct Access Register 1, DAR1

The Direct Access Registers provide a mechanism for directly mapping locations in the target memory system that is connected to the APB master interface.

Attributes

Offset

`0x0004`

Type

Read-write

Reset

`0x--------`

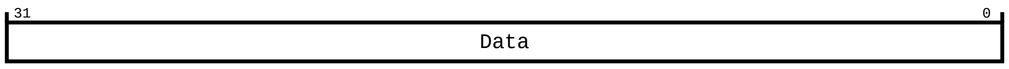
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-52: css600_ahbap_DAR1



The following table shows the bit assignments.

Table 10-55: DAR1 attributes

Bits	Reset value	Name	Type	Function
[31:0]	UNKNOWN	Data	Read-write	Maps to memory address ((TAR & `0xFFFFFC00`) + `0x4`). In read mode, the register contains the data value that was read from memory, and in write mode the register contains the data value to write to memory.

10.4.2.3 css600_ahbap Direct Access Register 2, DAR2

The Direct Access Registers provide a mechanism for directly mapping locations in the target memory system that is connected to the APB master interface.

Attributes

Offset

`0x0008`

Type

Read-write

Reset

`0x--------`

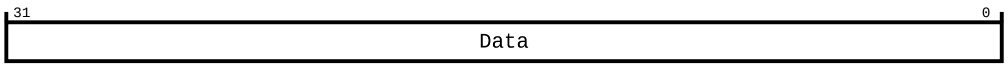
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-53: css600_ahbap_DAR2



The following table shows the bit assignments.

Table 10-56: DAR2 attributes

Bits	Reset value	Name	Type	Function
[31:0]	UNKNOWN	Data	Read-write	Maps to memory address ((TAR & `0xFFFFFC00`) + `0x8`). In read mode, the register contains the data value that was read from memory, and in write mode the register contains the data value to write to memory.

10.4.2.4 css600_ahbap Direct Access Register 255, DAR255

The Direct Access Registers provide a mechanism for directly mapping locations in the target memory system that is connected to the APB master interface.

Attributes

Offset

`0x03FC`

Type

Read-write

Reset

`0x--------`

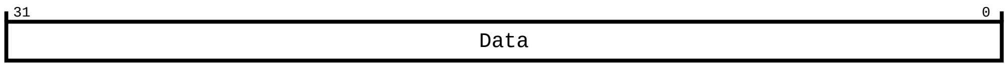
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-54: css600_ahbap_DAR255



The following table shows the bit assignments.

Table 10-57: DAR255 attributes

Bits	Reset value	Name	Type	Function
[31:0]	UNKNOWN	Data	Read-write	Maps to memory address ((TAR & `0xFFFFFC00`) + `0x3FC`). In read mode, the register contains the data value that was read from memory, and in write mode the register contains the data value to write to memory.

10.4.2.5 css600_ahbap Control Status Word register, CSW

The CSW register configures and controls accesses through the AHB master interface to the connected memory system.

Attributes

Offset

`0x0D00`

Type

Read-write

Reset

`0x43-000-2`

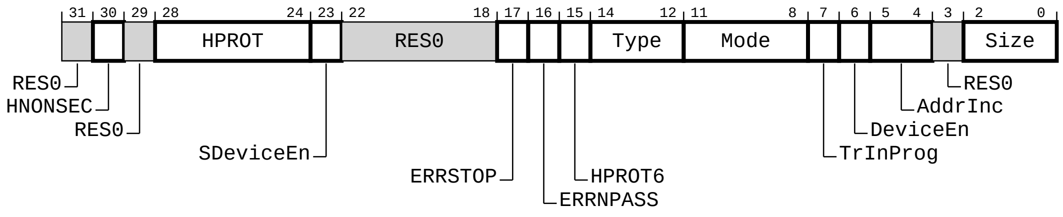
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-55: css600_ahbap_CSW



The following table shows the bit assignments.

Table 10-58: CSW attributes

Bits	Reset value	Name	Type	Function
[31]	`0b0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[30]	`0b1`	HNONSEC	Read-write	Drives hnonsec_m output pin. Together with the Access Port Enable interface signals, HNONSEC determines whether a secure access is allowed on the master interface. Accesses are permitted as follows, access_permitted = (ap_en && ap_secure_en) || (ap_en && HNONSEC).
[29]	`0b0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[28:24]	`0b00011`	HPROT	Read-write	Together with CSW.HPROT6, HPROT sets the protection control value to be output on hprot_m[6:0]. CSW.HPROT6 controls hprot_m[6]. CSW.HPROT controls hprot_m[4:0].hprot_m[5] is always driven LOW. This field is reset to `0x3`. The reset values of the two fields correspond to a protection value of: Non-Shareable, (Non-Allocate), Non-Lookup, Non-Modifiable, Non-Bufferable, Privileged, Data.css600_ahbap supports the following legal hprot_m encodings: • CSW.HPROT6=`0b0` CSW.HPROT[4:2]=`0b000`: Device-nE • CSW.HPROT6=`0b0` CSW.HPROT[4:2]=`0b001`: Device-E • CSW.HPROT6=`0b0` CSW.HPROT[4:2]=`0b010`: Normal Non-cacheable, Non-shareable • CSW.HPROT6=`0b0` CSW.HPROT[4:2]=`0b110`: Write-through, Non-shareable • CSW.HPROT6=`0b0` CSW.HPROT[4:2]=`0b111`: Write-back, Non-shareable • CSW.HPROT6=`0b1` CSW.HPROT[4:2]=`0b010`: Normal Non-cacheable, Shareable • CSW.HPROT6=`0b1` CSW.HPROT[4:2]=`0b110`: Write-through, Shareable • CSW.HPROT6=`0b1` CSW.HPROT[4:2]=`0b111`: Write-back, Shareable
[23]	UNKNOWN	SDeviceEn	Read-only	Indicates the status of the ap_en and ap_secure_en ports. It is set when both ap_en and ap_secure_en are HIGH, but otherwise is clear. If this bit is clear, Secure AHB transfers are not permitted. Non-secure memory accesses and internal register accesses that do not initiate memory accesses are permitted regardless of the status of this bit.
[22:18]	`0b00000`	RES0	Read-write	Reserved bit or field with SBZP behavior
[17]	`0b0`	ERRSTOP	Read-write	Stop on error: **0** Memory access errors do not prevent future memory accesses **1** Memory access errors prevent future memory accesses
[16]	`0b0`	ERRNPASS	Read-write	Errors are not passed upstream: **0** Memory access errors are passed upstream **1** Memory access errors are not passed upstream
[15]	`0b0`	HPROT6	Read-write	Together with CSW.HPROT, HPROT6 controls the protection value to be output on hprot_m[6:0].This field is reset to `0x0`.
[14:12]	`0b000`	Type	Read-only	This field is reserved. Reads return `0x0` and writes are ignored.
[11:8]	`0b0000`	Mode	Read-only	Specifies the mode of operation. All other values are reserved. **`0x0`** Normal download or upload mode
[7]	`0b0`	TrInProg	Read-only	Transfer in progress. This field indicates whether a transfer is in progress on the AHB master interface. If the master interface is busy, CSW.TrInProg is set in both logical APs.

Bits	Reset value	Name	Type	Function
[6]	UNKNOWN	DeviceEn	Read-only	Indicates the status of the ap_en port. The bit is set when ap_en is HIGH, and but otherwise is clear. If this bit is clear, no AHB transfers are carried out. That is, both secure and non-secure accesses are blocked.
[5:4]	`0b00`	AddrInc	Read-write	Auto address increment mode on RW data access. Only increments if the current transaction completes without an error response and the transaction is not aborted. **0x0** Auto increment OFF **0x1** Increment, single. Single transfer from corresponding byte lane **0x2** Reserved **0x3** Reserved
[3]	`0b0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[2:0]	`0b010`	Size	Read-write	Size of the data access to perform: **0x0** 8 bits **0x1** 16 bits **0x2** 32 bits **0x3** Reserved **0x4** Reserved **0x5** Reserved **0x6** Reserved **0x7** Reserved

10.4.2.6 css600_ahbap Transfer Address Register, TAR

TAR holds the transfer address of the current transfer. TAR must be programmed before initiating any memory transfer through DRW, or Banked Data Registers, or Direct Access Registers.

Attributes

Offset

`0x0D04`

Type

Read-write

Reset

`0x--------`

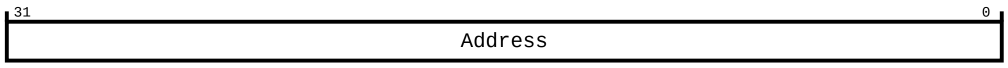
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-56: css600_ahbap_TAR



The following table shows the bit assignments.

Table 10-59: TAR attributes

Bits	Reset value	Name	Type	Function
[31:0]	UNKNOWN	Address	Read-write	Address of the current transfer. When a memory access is initiated by accessing the DRW register, the TAR value directly gives the 32-bit transfer address. When a memory access is initiated by accessing Banked Data registers, the TAR only provides the upper bits [31:4] and the remaining address bits [3:0] come from the offset of Banked Data register being accessed. When a memory access is initiated by accessing Direct Access Registers, the TAR provides the upper bits [31:10] and the remaining address bits [9:0] come from the offset of the DAR being accessed.

10.4.2.7 css600_ahbap Data Read/Write register, DRW

A write to the DRW register initiates a memory write transaction on the master. AP drives DRW write data on the data bus during the data phase of the current transfer. Reading the DRW register initiates a memory read transaction on the master. The resulting read data that is received from the memory system is returned on the slave interface.

Attributes

Offset

`0x0D0C`

Type

Read-write

Reset

`0x--------`

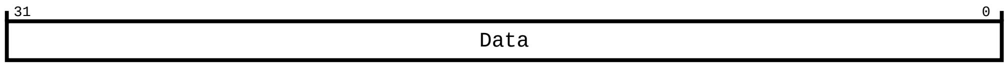
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-57: css600_ahbap_DRW



The following table shows the bit assignments.

Table 10-60: DRW attributes

Bits	Reset value	Name	Type	Function
[31:0]	UNKNOWN	Data	Read-write	Current transfer data value. In read mode, the register contains the data value that was read from the current transfer, and in write mode the register contains the data value to write for the current transfer.

10.4.2.8 css600_ahbap Banked Data register 0, BD0

The Banked Data registers provide a mechanism for directly mapping APB slave accesses to memory transfers without having to rewrite the TAR within a 16-byte boundary.

Attributes

Offset

`0x0D10`

Type

Read-write

Reset

`0x--------`

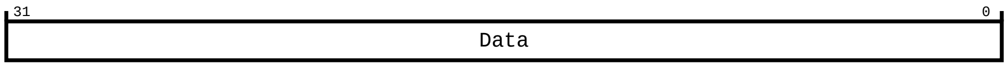
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-58: css600_ahbap_BD0



The following table shows the bit assignments.

Table 10-61: BD0 attributes

Bits	Reset value	Name	Type	Function
[31:0]	UNKNOWN	Data	Read-write	Maps to memory address ((TAR & `0xFFFFFFF0`) + `0x0`). In read mode, the register contains the data value that was read from the current transfer, and in write mode the register contains the data value to write for the current transfer.

10.4.2.9 css600_ahbap Banked Data register 1, BD1

The Banked Data registers provide a mechanism for directly mapping APB slave accesses to memory transfers without having to rewrite the TAR within a 16-byte boundary.

Attributes

Offset

`0x0D14`

Type

Read-write

Reset

`0x--------`

Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-59: css600_ahbap_BD1

The following table shows the bit assignments.

Table 10-62: BD1 attributes

Bits	Reset value	Name	Type	Function
[31:0]	UNKNOWN	Data	Read-write	Maps to memory address ((TAR & `0xFFFFFFF0`) + `0x4`). In read mode, the register contains the data value that was read from the current transfer, and in write mode the register contains the data value to write for the current transfer.

10.4.2.10 css600_ahbap Banked Data register 2, BD2

The Banked Data registers provide a mechanism for directly mapping APB slave accesses to memory transfers without having to rewrite the TAR within a 16-byte boundary.

Attributes

Offset

`0x0D18`

Type

Read-write

Reset

`0x--------`

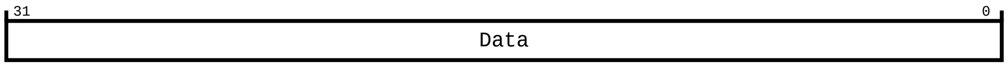
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-60: css600_ahbap_BD2



The following table shows the bit assignments.

Table 10-63: BD2 attributes

Bits	Reset value	Name	Type	Function
[31:0]	UNKNOWN	Data	Read-write	Maps to memory address ((TAR & `0xFFFFFFF0`) + `0x8`). In read mode, the register contains the data value that was read from the current transfer, and in write mode the register contains the data value to write for the current transfer.

10.4.2.11 css600_ahbap Banked Data register 3, BD3

The Banked Data registers provide a mechanism for directly mapping APB slave accesses to memory transfers without having to rewrite the TAR within a 16-byte boundary.

Attributes

Offset

`0x0D1C`

Type

Read-write

Reset

`0x--------`

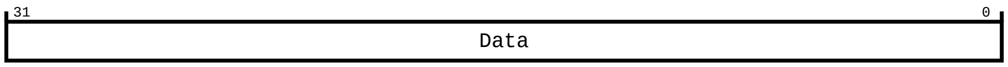
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-61: css600_ahbap_BD3



The following table shows the bit assignments.

Table 10-64: BD3 attributes

Bits	Reset value	Name	Type	Function
[31:0]	UNKNOWN	Data	Read-write	Maps to memory address ((TAR & `0xFFFFFFF0`) + `0xC`). In read mode, the register contains the data value that was read from the current transfer, and in write mode the register contains the data value to write for the current transfer.

10.4.2.12 css600_ahbap Transfer Response Register, TRR

The Transfer Response Register is used to capture an error response received during a transaction. It is also used to clear any logged responses.

Attributes

Offset

`0x0D24`

Type

Read-write

Reset

`0x00000000`

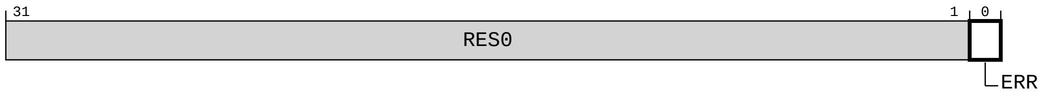
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-62: css600_ahbap_TRR



The following table shows the bit assignments.

Table 10-65: TRR attributes

Bits	Reset value	Name	Type	Function
[31:1]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior

Bits	Reset value	Name	Type	Function
[0]	`0b0`	ERR	Read-write	Logged error. **0** On reads - no error response logged. Writing to this bit has no effect. **1** On reads - error response logged. Writing to this bit clears this bit to 0.

10.4.2.13 css600_ahbap Configuration register, CFG

This is the css600_ahbap configuration register.

Attributes

Offset

`0x0DF4`

Type

Read-only

Reset

`0x000101A0`

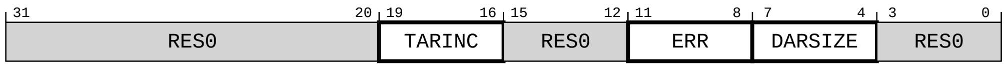
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-63: css600_ahbap_CFG



The following table shows the bit assignments.

Table 10-66: CFG attributes

Bits	Reset value	Name	Type	Function
[31:20]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[19:16]	`0b0001`	TARINC	Read-only	TAR incrementer size. Returns `0x1` indicating a TAR incrementer size of 10 bits.
[15:12]	`0b0000`	RES0	Read-only	Reserved bit or field with SBZP behavior
[11:8]	`0b0001`	ERR	Read-only	Error functionality implemented. Returns `0x1` indicating that Error Response Handling version 1 is implemented. See the Arm Debug Interface Architecture Specification ADIv6.0 for more information.
[7:4]	`0b1010`	DARSIZE	Read-only	Size of DAR register space. Returns `0xA` indicating that 1KB (256 registers, each 32 bits wide) of DAR is implemented.

Bits	Reset value	Name	Type	Function
[3:0]	`0b0000`	RES0	Read-only	Reserved bit or field with SBZP behavior

10.4.2.14 css600_ahbap Debug Base Address register, BASE

Provides an initial system address for the first component in the system. Typically, the system address is the address of a top-level ROM Table that indicates where APv2 APs are located.

Attributes

Offset

`0x0DF8`

Type

Read-only

Reset

`0x-----00-`

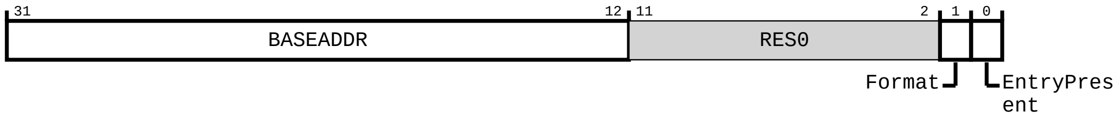
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-64: css600_ahbap_BASE



The following table shows the bit assignments.

Table 10-67: BASE attributes

Bits	Reset value	Name	Type	Function
[31:12]	IMPLEMENTATION DEFINED	BASEADDR	Read-only	Base address of a ROM table. It points to the start of the debug register space or a ROM table address. Bits[11:0] of the address are `0x000` because the address is aligned to 4KB boundary. This field is valid only if BASE.EntryPresent bit is set to 1, in which case it returns the tie-off value of the input signal baseaddr[31:12], otherwise, it reads as `0x0`.
[11:2]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[1]	`0b1`	Format	Read-only	Base address register format. Returns the value `0b1` indicating the ADIv5 format, which is unchanged in ADIv6.

Bits	Reset value	Name	Type	Function
[0]	**IMPLEMENTATION DEFINED**	EntryPresent	Read-only	This field indicates whether a debug component is present for this AP. It returns the tie-off value of the input signal baseaddr_valid. **0** No debug entry present. **1** Debug entry present and BASE.BASEADDR indicate the start address of the debug register space or ROM table.

10.4.2.15 css600_ahbap Identification Register, IDR

This register provides a mechanism for the debugger to know various identity attributes of the AP.

Attributes

Offset

`0x0DFC`

Type

Read-only

Reset

`0x44770008`

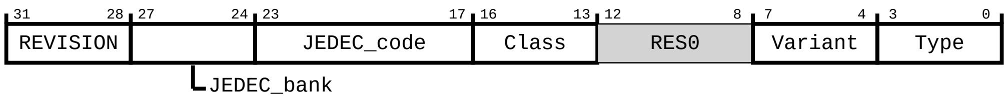
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-65: css600_ahbap_IDR



The following table shows the bit assignments.

Table 10-68: IDR attributes

Bits	Reset value	Name	Type	Function
[31:28]	`0b0100`	REVISION	Read-only	Revision. An incremental value starting at `0x0` for the first design of a component. See the Component list in Chapter 1 for information on the RTL revision of the component.
[27:24]	`0b0100`	JEDEC_bank	Read-only	The JEP106 continuation code. Returns `0x4`, indicating Arm as the designer.
[23:17]	`0x3B`	JEDEC_code	Read-only	The JEP106 identification code. Returns `0x3B`, indicating Arm as the designer.
[16:13]	`0b1000`	Class	Read-only	Returns `0x8`, indicating that this is a Memory Access Port

Bits	Reset value	Name	Type	Function
[12:8]	`0b00000`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0000`	Variant	Read-only	Returns `0x0`, indicating no variation from base type specified by IDR.Type
[3:0]	`0b1000`	Type	Read-only	Returns `0x8`, indicating that this is an AHB5 Access Port with full HPROT control

10.4.2.16 css600_ahbap Integration Test Status register, ITSTATUS

This register indicates the Integration Test DP Abort status.

Attributes

Offset

`0x0EFC`

Type

Read-write

Reset

`0x00000000`

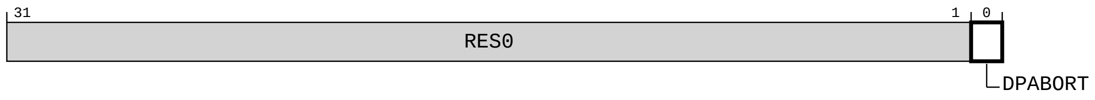
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-66: css600_ahbap_ITSTATUS



The following table shows the bit assignments.

Table 10-69: ITSTATUS attributes

Bits	Reset value	Name	Type	Function
[31:1]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[0]	`0b0`	DPABORT	Read-only	When in Integration testing mode (ITCTRL.IME=`0b1`): Behaves as a sticky bit and latches to 1 on a rising edge of dp_abort. Cleared on a read from this register. If dp_abort rises in the same cycle as a read of the ITSTATUS register is received, the read takes priority and the register is cleared. When in normal functional operation mode (ITCTRL.IME=`0b0`): Read as 0, writes ignored.

10.4.2.17 css600_ahbap Integration Mode Control Register, ITCTRL

The Integration Mode Control register is used to enable topology detection.

Attributes

Offset

`0x0F00`

Type

Read-write

Reset

`0x00000000`

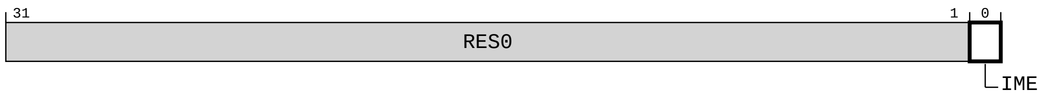
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-67: css600_ahbap_ITCTRL



The following table shows the bit assignments.

Table 10-70: ITCTRL attributes

Bits	Reset value	Name	Type	Function
[31:1]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[0]	`0b0`	IME	Read-write	Integration Mode Enable. When set, the component enters integration mode, enabling topology detection or integration testing to be performed.

10.4.2.18 css600_ahbap Claim Tag Set Register, CLAIMSET

This register forms one half of the claim tag value. On writes, this location enables individual bits to be set. On reads, it returns the number of bits that can be set.

Attributes

Offset

`0x0FA0`

Type

Read-write

Reset

`0x00000003`

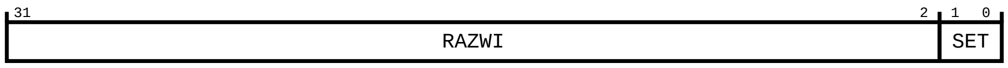
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-68: css600_ahbap_CLAIMSET



The following table shows the bit assignments.

Table 10-71: CLAIMSET attributes

Bits	Reset value	Name	Type	Function
[31:2]	`0x0`	RAZ/WI	Read-write	RAZ/WI
[1:0]	`0b11`	SET	Read-write	A bit-programmable register bank that sets the claim tag value. A read returns a logic 1 for all implemented locations.

10.4.2.19 css600_ahbap Claim Tag Clear Register, CLAIMCLR

This register forms one half of the claim tag value. On writes, this location enables individual bits to be cleared. On reads, it returns the current claim tag value.

Attributes

Offset

`0x0FA4`

Type

Read-write

Reset

`0x00000000`

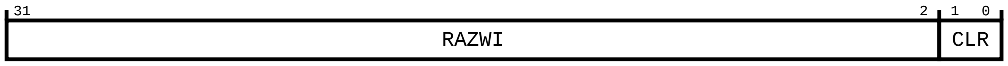
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-69: css600_ahbap_CLAIMCLR



The following table shows the bit assignments.

Table 10-72: CLAIMCLR attributes

Bits	Reset value	Name	Type	Function
[31:2]	`0x0`	RAZ/ WI	Read-write	RAZ/WI
[1:0]	`0b00`	CLR	Read-write	A bit-programmable register bank that clears the claim tag value. It is zero at reset. It is used by software agents to signal to each other ownership of the hardware. It has no direct effect on the hardware itself.

10.4.2.20 css600_ahbap Authentication Status Register, AUTHSTATUS

Reports the current status of the authentication control signals.

Attributes

Offset

`0x0FB8`

Type

Read-only

Reset

`0x000000--`

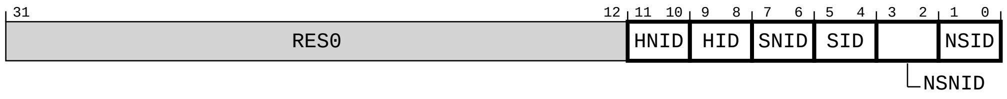
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-70: css600_ahbap_AUTHSTATUS



The following table shows the bit assignments.

Table 10-73: AUTHSTATUS attributes

Bits	Reset value	Name	Type	Function
[31:12]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[11:10]	`0b00`	HNID	Read-only	Hypervisor non-invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.
[9:8]	`0b00`	HID	Read-only	Hypervisor invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.
[7:6]	UNKNOWN	SNID	Read-only	Secure non-invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.
[5:4]	UNKNOWN	SID	Read-only	Secure invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.
[3:2]	UNKNOWN	NSNID	Read-only	Non-secure non-invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.
[1:0]	UNKNOWN	NSID	Read-only	Non-secure invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.

10.4.2.21 css600_ahbap Device Architecture Register, DEVARCH

Identifies the architect and architecture of a CoreSight component. The architect might differ from the designer of a component, for example Arm defines the architecture but another company designs and implements the component.

Attributes

Offset

`0x0FBC`

Type

Read-only

Reset

`0x47700A17`

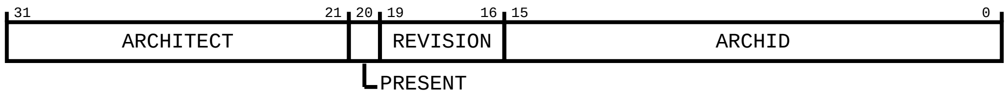
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-71: css600_ahbap_DEVARCH



The following table shows the bit assignments.

Table 10-74: DEVARCH attributes

Bits	Reset value	Name	Type	Function
[31:21]	`0x23B`	ARCHITECT	Read-only	Returns 0x23B, denoting Arm as architect of the component.
[20]	`0b1`	PRESENT	Read-only	Returns 1, indicating that the DEVARCH register is present.
[19:16]	`0b0000`	REVISION	Read-only	Architecture revision. Returns the revision of the architecture that the ARCHID field specifies.
[15:0]	`0xA17`	ARCHID	Read-only	Architecture ID. Returns 0x0A17, identifying APv2 MEM-AP architecture v0.

10.4.2.22 css600_ahbap Device Type Identifier Register, DEVTYPE

A debugger can use this register to get information about a component that has an unrecognized Part number.

Attributes

Offset

`0x0FCC`

Type

Read-only

Reset

`0x00000000`

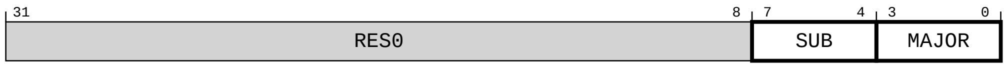
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-72: css600_ahbap_DEVTYPE



The following table shows the bit assignments.

Table 10-75: DEVTYPE attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0000`	SUB	Read-only	Minor classification. Returns `0x0`, Other/undefined.
[3:0]	`0b0000`	MAJOR	Read-only	Major classification. Returns `0x0`, Miscellaneous.

10.4.2.23 css600_ahbap Peripheral Identification Register 4, PIDR4

The PIDR4 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FD0`

Type

Read-only

Reset

`0x00000004`

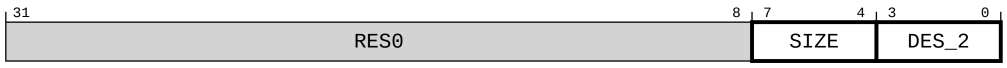
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-73: css600_ahbap_PIDR4



The following table shows the bit assignments.

Table 10-76: PIDR4 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0000`	SIZE	Read-only	Indicates the memory size that is used by this component. Returns 0 indicating that the component uses an **UNKNOWN** number of 4KB blocks. Using the SIZE field to indicate the size of the component is deprecated.
[3:0]	`0b0100`	DES_2	Read-only	JEP106 continuation code. Together, with PIDR2.DES_1 and PIDR1.DES_0, they indicate the designer of the component and not the implementer, except where the two are the same.

10.4.2.24 css600_ahbap Peripheral Identification Register 5, PIDR5

The PIDR5 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FD4`

Type

Read-only

Reset

`0x00000000`

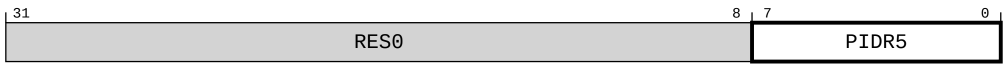
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-74: css600_ahbap_PIDR5



The following table shows the bit assignments.

Table 10-77: PIDR5 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	PIDR5	Read-only	Reserved.

10.4.2.25 css600_ahbap Peripheral Identification Register 6, PIDR6

The PIDR6 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FD8`

Type

Read-only

Reset

`0x00000000`

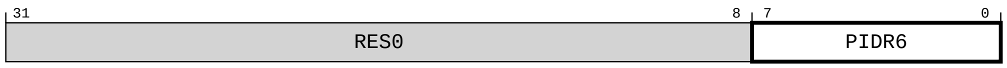
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-75: css600_ahbap_PIDR6



The following table shows the bit assignments.

Table 10-78: PIDR6 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	PIDR6	Read-only	Reserved.

10.4.2.26 css600_ahbap Peripheral Identification Register 7, PIDR7

The PIDR7 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FDC`

Type

Read-only

Reset

`0x00000000`

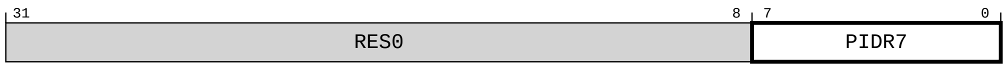
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-76: css600_ahbap_PIDR7



The following table shows the bit assignments.

Table 10-79: PIDR7 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	PIDR7	Read-only	Reserved.

10.4.2.27 css600_ahbap Peripheral Identification Register 0, PIDR0

The PIDR0 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FE0`

Type

Read-only

Reset

`0x000000E3`

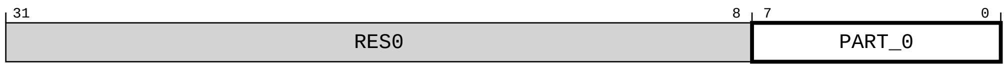
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-77: css600_ahbap_PIDR0



The following table shows the bit assignments.

Table 10-80: PIDR0 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	0xE3	PART_0	Read-only	Part number, bits[7:0]. Taken together with PIDR1.PART_1 it indicates the component. The Part Number is selected by the designer of the component.

10.4.2.28 css600_ahbap Peripheral Identification Register 1, PIDR1

The PIDR1 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FE4`

Type

Read-only

Reset

`0x000000B9`

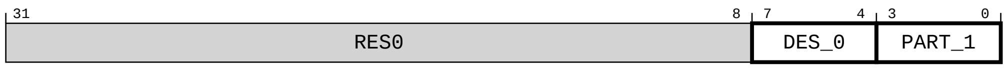
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-78: css600_ahbap_PIDR1



The following table shows the bit assignments.

Table 10-81: PIDR1 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b1011`	DES_0	Read-only	JEP106 identification code, bits[3:0]. Together, with PIDR4.DES_2 and PIDR2.DES_1, they indicate the designer of the component and not the implementer, except where the two are the same.
[3:0]	`0b1001`	PART_1	Read-only	Part number, bits[11:8]. Taken together with PIDR0.PART_0 it indicates the component. The Part Number is selected by the designer of the component.

10.4.2.29 css600_ahbap Peripheral Identification Register 2, PIDR2

The PIDR2 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FE8`

Type

Read-only

Reset

`0x0000004B`

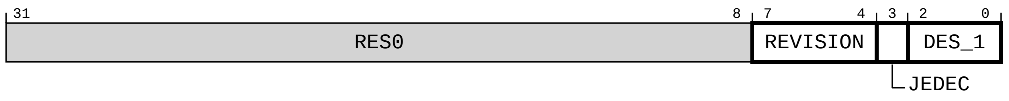
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-79: css600_ahbap_PIDR2



The following table shows the bit assignments.

Table 10-82: PIDR2 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0100`	REVISION	Read-only	Revision. It is an incremental value starting at `0x0` for the first design of a component. See the Component list in Chapter 1 for information on the RTL revision of the component.
[3]	`0b1`	JEDEC	Read-only	1 - Always set. Indicates that a JEDEC assigned value is used.
[2:0]	`0b011`	DES_1	Read-only	JEP106 identification code, bits[6:4]. Together, with PIDR4.DES_2 and PIDR1.DES_0, they indicate the designer of the component and not the implementer, except where the two are the same.

10.4.2.30 css600_ahbap Peripheral Identification Register 3, PIDR3

The PIDR3 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FEC`

Type

Read-only

Reset

`0x00000000`

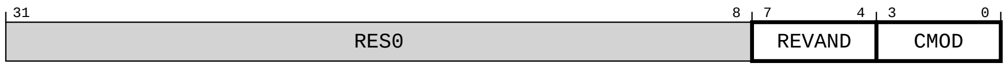
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-80: css600_ahbap_PIDR3



The following table shows the bit assignments.

Table 10-83: PIDR3 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0000`	REVAND	Read-only	This field indicates minor errata fixes specific to this design, for example metal fixes after implementation. In most cases this field is `0x0`.
[3:0]	`0b0000`	CMOD	Read-only	Customer Modified. Where the component is reusable IP, this value indicates if the customer has modified the behavior of the component. In most cases this field is `0x0`.

10.4.2.31 css600_ahbap Component Identification Register 0, CIDR0

The CIDR0 register is part of the set of component identification registers.

Attributes

Offset

`0x0FF0`

Type

Read-only

Reset

`0x0000000D`

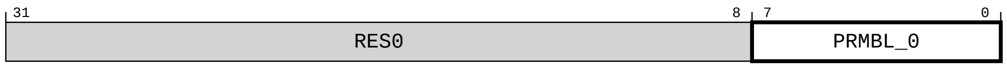
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-81: css600_ahbap_CIDR0



The following table shows the bit assignments.

Table 10-84: CIDR0 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0xD`	PRMBL_0	Read-only	Preamble. Returns `0x0D`.

10.4.2.32 css600_ahbap Component Identification Register 1, CIDR1

The CIDR1 register is part of the set of component identification registers.

Attributes

Offset

`0x0FF4`

Type

Read-only

Reset

`0x00000090`

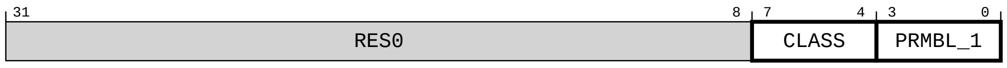
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-82: css600_ahbap_CIDR1



The following table shows the bit assignments.

Table 10-85: CIDR1 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b1001`	CLASS	Read-only	Component class. Returns `0x9`, indicating this is a CoreSight component.
[3:0]	`0b0000`	PRMBL_1	Read-only	Preamble. Returns `0x0`.

10.4.2.33 css600_ahbap Component Identification Register 2, CIDR2

The CIDR2 register is part of the set of component identification registers.

Attributes

Offset

`0x0FF8`

Type

Read-only

Reset

`0x00000005`

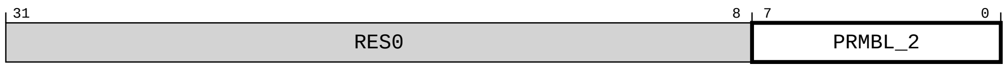
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-83: css600_ahbap_CIDR2



The following table shows the bit assignments.

Table 10-86: CIDR2 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x5`	PRMBL_2	Read-only	Preamble. Returns `0x05`.

10.4.2.34 css600_ahbap Component Identification Register 3, CIDR3

The CIDR3 register is part of the set of component identification registers.

Attributes

Offset

`0x0FFC`

Type

Read-only

Reset

`0x000000B1`

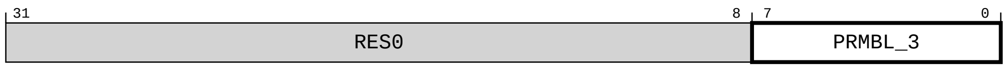
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-84: css600_ahbap_CIDR3



The following table shows the bit assignments.

Table 10-87: CIDR3 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0xB1`	PRMBL_3	Read-only	Preamble. Returns `0xB1`.

10.5 `css600_apv1adapter` introduction

This section describes the programmers model of the `css600_apv1adapter`.

10.5.1 css600_apv1adapter register summary

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

Note

A reset value containing one or more '-' means that this register contains **UNKNOWN** or **IMPLEMENTATION-DEFINED** values. See the relevant register description for more information.

Locations that are not listed in the table are Reserved.

Table 10-88: css600_apv1adapter register summary

Offset	Name	Type	Reset	Width	Description
0x0D00	Downstream reg 0	IMPLEMENTATION-DEFINED	0x--------	32	Accesses APv1 register at 0x00000D00
0x0D04	Downstream reg 1	IMPLEMENTATION-DEFINED	0x--------	32	Accesses APv1 register at 0x00000D04
0x0D08	Downstream reg 2	IMPLEMENTATION-DEFINED	0x--------	32	Accesses APv1 register at 0x00000D08
...	...	...	...	...	...
0x0DFC	Downstream reg 63	IMPLEMENTATION-DEFINED	0x--------	32	Accesses APv1 register at 0x00000DFC
0x0EFC	ITSTATUS	RW	0x00000000	32	css600_apv1adapter Integration Test Status register, ITSTATUS
0x0F00	ITCTRL	RW	0x00000000	32	css600_apv1adapter Integration Mode Control Register, ITCTRL
0x0FA0	CLAIMSET	RW	0x00000003	32	css600_apv1adapter Claim Tag Set Register, CLAIMSET
0x0FA4	CLAIMCLR	RW	0x00000000	32	css600_apv1adapter Claim Tag Clear Register, CLAIMCLR
0x0FB8	AUTHSTATUS	RO	0x00000000	32	css600_apv1adapter Authentication Status Register, AUTHSTATUS
0x0FBC	DEVARCH	RO	0x47700A47	32	css600_apv1adapter Device Architecture Register, DEVARCH
0x0FCC	DEVTYPE	RO	0x00000000	32	css600_apv1adapter Device Type Identifier Register, DEVTYPE
0x0FD0	PIDR4	RO	0x00000004	32	css600_apv1adapter Peripheral Identification Register 4, PIDR4
0x0FD4	PIDR5	RO	0x00000000	32	css600_apv1adapter Peripheral Identification Register 5, PIDR5
0x0FD8	PIDR6	RO	0x00000000	32	css600_apv1adapter Peripheral Identification Register 6, PIDR6
0x0FDC	PIDR7	RO	0x00000000	32	css600_apv1adapter Peripheral Identification Register 7, PIDR7
0x0FE0	PIDR0	RO	0x000000E5	32	css600_apv1adapter Peripheral Identification Register 0, PIDR0
0x0FE4	PIDR1	RO	0x000000B9	32	css600_apv1adapter Peripheral Identification Register 1, PIDR1
0x0FE8	PIDR2	RO	0x0000000B	32	css600_apv1adapter Peripheral Identification Register 2, PIDR2
0x0FEC	PIDR3	RO	0x00000000	32	css600_apv1adapter Peripheral Identification Register 3, PIDR3
0x0FF0	CIDR0	RO	0x0000000D	32	css600_apv1adapter Component Identification Register 0, CIDR0
0x0FF4	CIDR1	RO	0x00000090	32	css600_apv1adapter Component Identification Register 1, CIDR1
0x0FF8	CIDR2	RO	0x00000005	32	css600_apv1adapter Component Identification Register 2, CIDR2

Offset	Name	Type	Reset	Width	Description
`0x0FFC`	CIDR3	RO	`0x000000B1`	32	css600_apv1adapter Component Identification Register 3, CIDR3

10.5.2 Register descriptions

This section describes the `css600_apv1adapter` registers.

css600_apv1adapter register summary provides cross references to individual registers.

10.5.2.1 css600_apv1adapter Integration Test Status register, ITSTATUS

This register indicates the Integration Test DP Abort status.

Attributes

Offset

`0x0EFC`

Type

Read-write

Reset

`0x00000000`

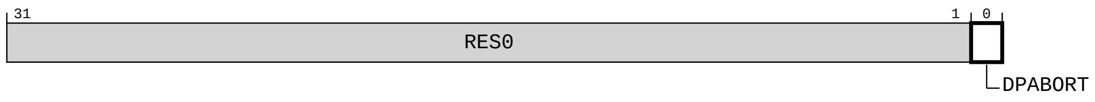
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-85: css600_apv1adapter_ITSTATUS



The following table shows the bit assignments.

Table 10-89: ITSTATUS attributes

Bits	Reset value	Name	Type	Function
[31:1]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior

Bits	Reset value	Name	Type	Function
[0]	`0b0`	DPABORT	Read-only	When in Integration testing mode (ITCTRL.IME=`0b1`): Behaves as a sticky bit and latches to 1 on a rising edge of dp_abort. Cleared on a read from this register. If dp_abort rises in the same cycle as a read of the ITSTATUS register is received, the read takes priority and the register is cleared. When in normal functional operation mode (ITCTRL.IME=`0b0`): Read as 0, writes ignored.

10.5.2.2 css600_apv1adapter Integration Mode Control Register, ITCTRL

The Integration Mode Control register is used to enable topology detection.

Attributes

Offset

`0x0F00`

Type

Read-write

Reset

`0x00000000`

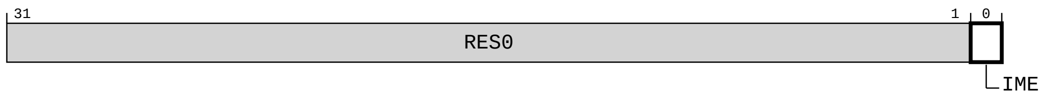
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-86: css600_apv1adapter_ITCTRL



The following table shows the bit assignments.

Table 10-90: ITCTRL attributes

Bits	Reset value	Name	Type	Function
[31:1]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[0]	`0b0`	IME	Read-write	Integration Mode Enable. When set, the component enters integration mode, enabling topology detection or integration testing to be performed.

10.5.2.3 css600_apv1adapter Claim Tag Set Register, CLAIMSET

This register forms one half of the claim tag value. On writes, this location enables individual bits to be set. On reads, it returns the number of bits that can be set.

Attributes

Offset

`0x0FA0`

Type

Read-write

Reset

`0x00000003`

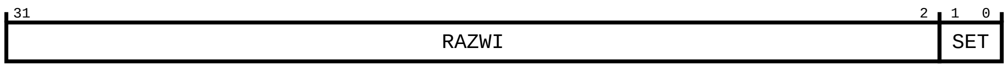
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-87: css600_apv1adapter_CLAIMSET



The following table shows the bit assignments.

Table 10-91: CLAIMSET attributes

Bits	Reset value	Name	Type	Function
[31:2]	`0x0`	RAZ/WI	Read-write	RAZ/WI
[1:0]	`0b11`	SET	Read-write	A bit-programmable register bank that sets the claim tag value. A read returns a logic 1 for all implemented locations.

10.5.2.4 css600_apv1adapter Claim Tag Clear Register, CLAIMCLR

This register forms one half of the claim tag value. On writes, this location enables individual bits to be cleared. On reads, it returns the current claim tag value.

Attributes

Offset

`0x0FA4`

Type

Read-write

Reset

`0x00000000`

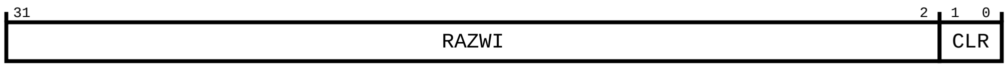
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-88: css600_apv1adapter_CLAIMCLR



The following table shows the bit assignments.

Table 10-92: CLAIMCLR attributes

Bits	Reset value	Name	Type	Function
[31:2]	`0x0`	RAZ/WI	Read-write	RAZ/WI
[1:0]	`0b00`	CLR	Read-write	A bit-programmable register bank that clears the claim tag value. It is zero at reset. It is used by software agents to signal to each other ownership of the hardware. It has no direct effect on the hardware itself.

10.5.2.5 css600_apv1adapter Authentication Status Register, AUTHSTATUS

Reports the current status of the authentication control signals.

Attributes

Offset

`0x0FB8`

Type

Read-only

Reset

`0x00000000`

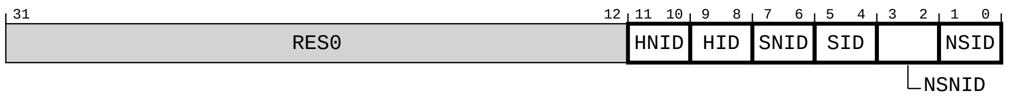
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-89: css600_apv1adapter_AUTHSTATUS



The following table shows the bit assignments.

Table 10-93: AUTHSTATUS attributes

Bits	Reset value	Name	Type	Function
[31:12]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[11:10]	`0b00`	HNID	Read-only	Hypervisor non-invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.
[9:8]	`0b00`	HID	Read-only	Hypervisor invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.
[7:6]	`0b00`	SNID	Read-only	Secure non-invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.
[5:4]	`0b00`	SID	Read-only	Secure invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.
[3:2]	`0b00`	NSNID	Read-only	Non-secure non-invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.
[1:0]	`0b00`	NSID	Read-only	Non-secure invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.

10.5.2.6 css600_apv1adapter Device Architecture Register, DEVARCH

Identifies the architect and architecture of a CoreSight component. The architect might differ from the designer of a component, for example Arm defines the architecture but another company designs and implements the component.

Attributes

Offset

`0x0FBC`

Type

Read-only

Reset

`0x47700A47`

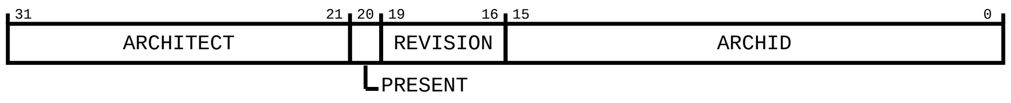
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-90: css600_apv1adapter_DEVARCH



The following table shows the bit assignments.

Table 10-94: DEVARCH attributes

Bits	Reset value	Name	Type	Function
[31:21]	`0x23B`	ARCHITECT	Read-only	Returns 0x23B, denoting Arm as architect of the component
[20]	`0b1`	PRESENT	Read-only	Returns 1, indicating that the DEVARCH register is present
[19:16]	`0b0000`	REVISION	Read-only	Architecture revision. Returns the revision of the architecture that the ARCHID field specifies.
[15:0]	`0xA47`	ARCHID	Read-only	Architecture ID. Returns 0x0A47, identifying Unknown AP.

10.5.2.7 css600_apv1adapter Device Type Identifier Register, DEVTYPE

A debugger can use this register to get information about a component that has an unrecognized Part number.

Attributes

Offset

`0x0FCC`

Type

Read-only

Reset

`0x00000000`

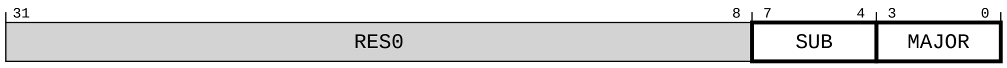
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-91: css600_apv1adapter_DEVTYPE



The following table shows the bit assignments.

Table 10-95: DEVTYPE attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0000`	SUB	Read-only	Minor classification. Returns `0x0`, Other/undefined.
[3:0]	`0b0000`	MAJOR	Read-only	Major classification. Returns `0x0`, Miscellaneous.

10.5.2.8 css600_apv1adapter Peripheral Identification Register 4, PIDR4

The PIDR4 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FD0`

Type

Read-only

Reset

`0x00000004`

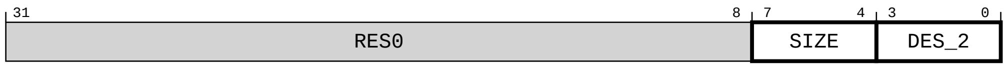
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-92: css600_apv1adapter_PIDR4



The following table shows the bit assignments.

Table 10-96: PIDR4 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0000`	SIZE	Read-only	Indicates the memory size that is used by this component. Returns 0 indicating that the component uses an **UNKNOWN** number of 4KB blocks. Using the SIZE field to indicate the size of the component is deprecated.
[3:0]	`0b0100`	DES_2	Read-only	JEP106 continuation code. Together, with PIDR2.DES_1 and PIDR1.DES_0, they indicate the designer of the component and not the implementer, except where the two are the same.

10.5.2.9 css600_apv1adapter Peripheral Identification Register 5, PIDR5

The PIDR5 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FD4`

Type

Read-only

Reset

`0x00000000`

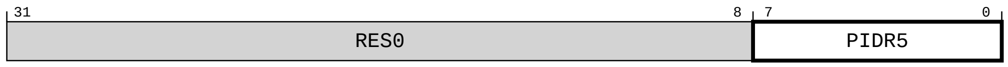
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-93: css600_apv1adapter_PIDR5



The following table shows the bit assignments.

Table 10-97: PIDR5 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	PIDR5	Read-only	Reserved.

10.5.2.10 css600_apv1adapter Peripheral Identification Register 6, PIDR6

The PIDR6 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FD8`

Type

Read-only

Reset

`0x00000000`

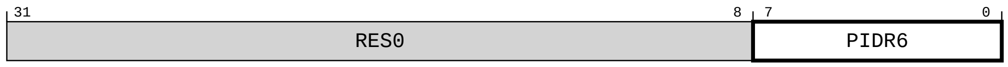
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-94: css600_apv1adapter_PIDR6



The following table shows the bit assignments.

Table 10-98: PIDR6 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	PIDR6	Read-only	Reserved.

10.5.2.11 css600_apv1adapter Peripheral Identification Register 7, PIDR7

The PIDR7 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FDC`

Type

Read-only

Reset

`0x00000000`

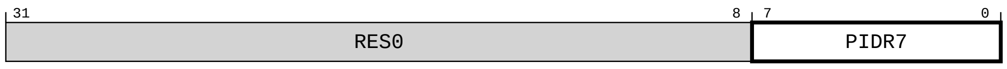
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-95: css600_apv1adapter_PIDR7



The following table shows the bit assignments.

Table 10-99: PIDR7 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	PIDR7	Read-only	Reserved.

10.5.2.12 css600_apv1adapter Peripheral Identification Register 0, PIDR0

The PIDR0 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FE0`

Type

Read-only

Reset

`0x000000E5`

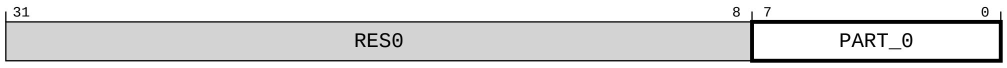
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-96: css600_apv1adapter_PIDR0



The following table shows the bit assignments.

Table 10-100: PIDR0 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	0xE5	PART_0	Read-only	Part number, bits[7:0]. Taken together with PIDR1.PART_1 it indicates the component. The Part Number is selected by the designer of the component.

10.5.2.13 css600_apv1adapter Peripheral Identification Register 1, PIDR1

The PIDR1 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FE4`

Type

Read-only

Reset

`0x000000B9`

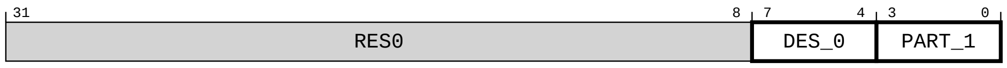
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-97: css600_apv1adapter_PIDR1



The following table shows the bit assignments.

Table 10-101: PIDR1 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b1011`	DES_0	Read-only	JEP106 identification code, bits[3:0]. Together, with PIDR4.DES_2 and PIDR2.DES_1, they indicate the designer of the component and not the implementer, except where the two are the same.
[3:0]	`0b1001`	PART_1	Read-only	Part number, bits[11:8]. Taken together with PIDR0.PART_0 it indicates the component. The Part Number is selected by the designer of the component.

10.5.2.14 css600_apv1adapter Peripheral Identification Register 2, PIDR2

The PIDR2 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FE8`

Type

Read-only

Reset

`0x0000000B`

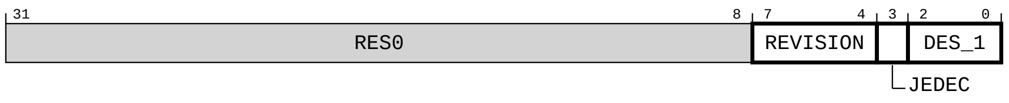
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-98: css600_apv1adapter_PIDR2



The following table shows the bit assignments.

Table 10-102: PIDR2 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0000`	REVISION	Read-only	Revision. It is an incremental value starting at `0x0` for the first design of a component. See the Component list in Chapter 1 for information on the RTL revision of the component.

Bits	Reset value	Name	Type	Function
[3]	0b1	JEDEC	Read-only	1 - Always set. Indicates that a JEDEC assigned value is used.
[2:0]	0b011	DES_1	Read-only	JEP106 identification code, bits[6:4]. Together, with PIDR4.DES_2 and PIDR1.DES_0, they indicate the designer of the component and not the implementer, except where the two are the same.

10.5.2.15 css600_apv1adapter Peripheral Identification Register 3, PIDR3

The PIDR3 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FEC`

Type

Read-only

Reset

`0x00000000`

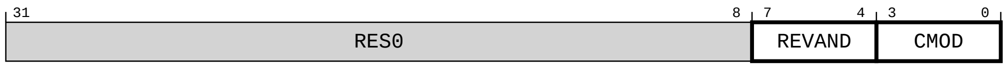
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-99: css600_apv1adapter_PIDR3



The following table shows the bit assignments.

Table 10-103: PIDR3 attributes

Bits	Reset value	Name	Type	Function
[31:8]	0x0	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	0b0000	REVAND	Read-only	This field indicates minor errata fixes specific to this design, for example metal fixes after implementation. In most cases this field is `0x0`.
[3:0]	0b0000	CMOD	Read-only	Customer Modified. Where the component is reusable IP, this value indicates if the customer has modified the behavior of the component. In most cases this field is `0x0`.

10.5.2.16 css600_apv1adapter Component Identification Register 0, CIDR0

The CIDR0 register is part of the set of component identification registers.

Attributes

Offset

`0x0FF0`

Type

Read-only

Reset

`0x0000000D`

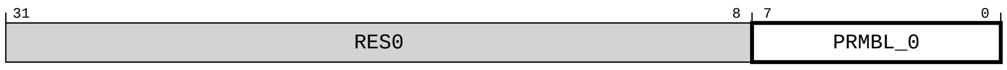
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-100: css600_apv1adapter_CIDR0



The following table shows the bit assignments.

Table 10-104: CIDR0 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0xD`	PRMBL_0	Read-only	Preamble. Returns `0x0D`.

10.5.2.17 css600_apv1adapter Component Identification Register 1, CIDR1

The CIDR1 register is part of the set of component identification registers.

Attributes

Offset

`0x0FF4`

Type

Read-only

Reset

`0x00000090`

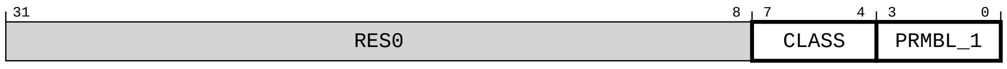
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-101: css600_apv1adapter_CIDR1



The following table shows the bit assignments.

Table 10-105: CIDR1 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b1001`	CLASS	Read-only	Component class. Returns `0x9`, indicating this is a CoreSight component.
[3:0]	`0b0000`	PRMBL_1	Read-only	Preamble. Returns `0x0`.

10.5.2.18 css600_apv1adapter Component Identification Register 2, CIDR2

The CIDR2 register is part of the set of component identification registers.

Attributes

Offset

`0x0FF8`

Type

Read-only

Reset

`0x00000005`

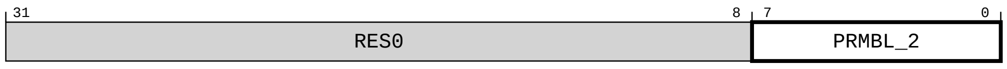
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-102: css600_apv1adapter_CIDR2



The following table shows the bit assignments.

Table 10-106: CIDR2 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x5`	PRMBL_2	Read-only	Preamble. Returns `0x05`.

10.5.2.19 css600_apv1adapter Component Identification Register 3, CIDR3

The CIDR3 register is part of the set of component identification registers.

Attributes

Offset

`0x0FFC`

Type

Read-only

Reset

`0x000000B1`

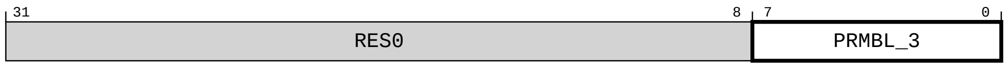
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-103: css600_apv1adapter_CIDR3



The following table shows the bit assignments.

Table 10-107: CIDR3 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0xB1`	PRMBL_3	Read-only	Preamble. Returns `0xB1`.

10.6 `css600_jtagap` introduction

This section describes the programmers model of the `css600_jtagap`.

10.6.1 css600_jtagap register summary

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

Note

A reset value containing one or more '-' means that this register contains **UNKNOWN** or **IMPLEMENTATION-DEFINED** values. See the relevant register description for more information.

Locations that are not listed in the table are Reserved.

Table 10-108: css600_jtagap register summary

Offset	Name	Type	Reset	Width	Description
`0x0D00`	CSW	RW	`0x00000000`	32	css600_jtagap Control/Status Word register, CSW
`0x0D04`	PSEL	RW	`0x00000000`	32	css600_jtagap Port Select register, PSEL
`0x0D08`	PSTA	RW	`0x00000000`	32	css600_jtagap Port Status register, PSTA
`0x0D10`	BFIFO1	RW	`0x000000--`	32	css600_jtagap Byte FIFO Registers, BFIFO1
`0x0D14`	BFIFO2	RW	`0x0000----`	32	css600_jtagap Byte FIFO Registers, BFIFO2
`0x0D18`	BFIFO3	RW	`0x00------`	32	css600_jtagap Byte FIFO Registers, BFIFO3
`0x0D1C`	BFIFO4	RW	`0x--------`	32	css600_jtagap Byte FIFO Registers, BFIFO4
`0x0DFC`	IDR	RO	`0x34760020`	32	css600_jtagap Identification Register, IDR
`0x0EFC`	ITSTATUS	RW	`0x00000000`	32	css600_jtagap Integration Test Status register, ITSTATUS
`0x0F00`	ITCTRL	RW	`0x00000000`	32	css600_jtagap Integration Mode Control Register, ITCTRL
`0x0FA0`	CLAIMSET	RW	`0x00000003`	32	css600_jtagap Claim Tag Set Register, CLAIMSET
`0x0FA4`	CLAIMCLR	RW	`0x00000000`	32	css600_jtagap Claim Tag Clear Register, CLAIMCLR
`0x0FBC`	DEVARCH	RO	`0x47700A27`	32	css600_jtagap Device Architecture Register, DEVARCH
`0x0FCC`	DEVTYPE	RO	`0x00000000`	32	css600_jtagap Device Type Identifier Register, DEVTYPE
`0x0FD0`	PIDR4	RO	`0x00000004`	32	css600_jtagap Peripheral Identification Register 4, PIDR4
`0x0FD4`	PIDR5	RO	`0x00000000`	32	css600_jtagap Peripheral Identification Register 5, PIDR5
`0x0FD8`	PIDR6	RO	`0x00000000`	32	css600_jtagap Peripheral Identification Register 6, PIDR6
`0x0FDC`	PIDR7	RO	`0x00000000`	32	css600_jtagap Peripheral Identification Register 7, PIDR7
`0x0FE0`	PIDR0	RO	`0x000000E6`	32	css600_jtagap Peripheral Identification Register 0, PIDR0
`0x0FE4`	PIDR1	RO	`0x000000B9`	32	css600_jtagap Peripheral Identification Register 1, PIDR1
`0x0FE8`	PIDR2	RO	`0x0000003B`	32	css600_jtagap Peripheral Identification Register 2, PIDR2
`0x0FEC`	PIDR3	RO	`0x00000000`	32	css600_jtagap Peripheral Identification Register 3, PIDR3
`0x0FF0`	CIDR0	RO	`0x0000000D`	32	css600_jtagap Component Identification Register 0, CIDR0
`0x0FF4`	CIDR1	RO	`0x00000090`	32	css600_jtagap Component Identification Register 1, CIDR1
`0x0FF8`	CIDR2	RO	`0x00000005`	32	css600_jtagap Component Identification Register 2, CIDR2
`0x0FFC`	CIDR3	RO	`0x000000B1`	32	css600_jtagap Component Identification Register 3, CIDR3

10.6.2 Register descriptions

This section describes the `css600_jtagap` registers.

css600_jtagap register summary provides cross references to individual registers.

10.6.2.1 css600_jtagap Control/Status Word register, CSW

The CSW register configures and controls transfers through the JTAG interface to the connected memory system.

Attributes

Offset

`0x0D00`

Type

Read-write

Reset

`0x00000000`

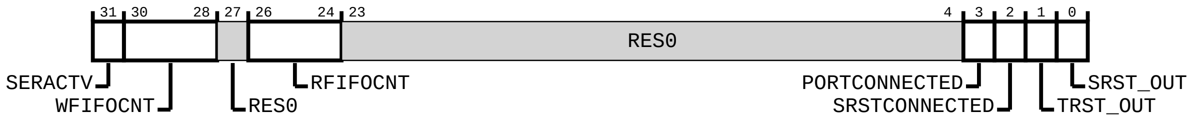
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-104: css600_jtagap_CSW



The following table shows the bit assignments.

Table 10-109: CSW attributes

Bits	Reset value	Name	Type	Function
[31]	`0b0`	SERACTV	Read-only	JTAG engine active. This bit gets set when the JTAG engine picks the first command from the Command FIFO for execution and remains set until all commands have been executed, that is until after CSW.WFIFOCNT becomes 0 and the JTAG engine goes to idle state. **0** JTAG engine is inactive. **1** JTAG engine is processing commands from the Command FIFO.

Bits	Reset value	Name	Type	Function
[30:28]	`0b000`	WFIFOCNT	Read-only	Command FIFO outstanding byte count. The reset value is `0x0`. Returns the number of command bytes held in the Command FIFO that are yet to be processed by the JTAG engine. Since the Command FIFO is 4 entries deep, this field can only take values between 0 and 4.
[27]	`0b0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[26:24]	`0b000`	RFIFOCNT	Read-only	Response FIFO outstanding byte count. The reset value is `0x0`. Returns the number of bytes of response data held in the Response FIFO. Since the Response FIFO is 7 entries deep, this field can take any value between 0 and 7.
[23:4]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[3]	`0b0`	PORTCONNECTED	Read-only	PORT connected. This bit indicates the logical AND of port_connected inputs from all ports that are currently selected in the PSEL register.
[2]	`0b0`	SRSTCONNECTED	Read-only	SRST connected. This bit is logical AND of srst_connected inputs from all ports that are currently selected in PSEL register.
[1]	`0b0`	TRST_OUT	Read-write	This bit specifies the value to drive out on the active-LOW cs_ntrst pin for the ports that are connected, selected, and their PSTA bit is clear. This bit does not self-clear and must be cleared by a software write to this register. **0** De-assert cs_ntrst HIGH. **1** Assert cs_ntrst LOW.
[0]	`0b0`	SRST_OUT	Read-write	This bit specifies the value to drive out on the active-LOW srst_out_n pin for the ports that are connected, selected, and their PSTA bit is clear. This bit does not self-clear and must be cleared by a software write to this register. **0** De-assert srst_out_n HIGH. **1** Assert srst_out_n LOW.

10.6.2.2 css600_jtagap Port Select register, PSEL

The PSEL register enables JTAG ports if the slave interface is connected to the JTAG AP and the port_enabled signal from the slave interface to the JTAG AP is asserted HIGH. The PSEL register must be written only when the following conditions are met: the JTAG engine is idle and the write FIFO is empty. If this register is written to in any other state, the corresponding JTAG ports are abruptly enabled, or disabled, in the middle of a transfer, which might cause errors, stalls, or deadlocks in the JTAG slave.

Attributes

Offset

`0x0D04`

Type

Read-write

Reset

`0x00000000`

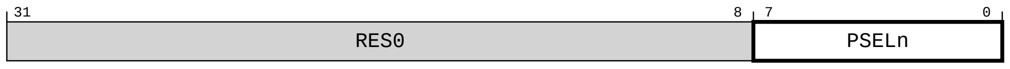
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-105: css600_jtagap_PSEL



The following table shows the bit assignments.

Table 10-110: PSEL attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	PSELn	Read-write	Port Select. Each register field is named as PSELn, where n = 0-7. The numerical index represents the bit position of that field in this register.

10.6.2.3 css600_jtagap Port Status register, PSTA

The PSTA register captures the state of a connected and selected port on every clock cycle. If a connected and selected port is disabled or powered down, that is the signal port_enabled goes LOW, even transiently, the corresponding bit in the PSTA register is set in the next cycle. It remains 1 until it is cleared by writing 1 to it. It gets cleared automatically on abort. Deselecting a port in PSEL does not alter the state of PSTA. If the PSTA bit is set for a port, that port is disabled and its TCK, TMS, and TDI outputs are driven LOW until its PSTA bit is cleared. Software must not clear any PSTA bit unless the JTAG-AP is idle, that is CSW.SERACTV=0b0 and CSW.WFIFOCNT=`0x0`.

Attributes

Offset

`0x0D08`

Type

Read-write

Reset

`0x00000000`

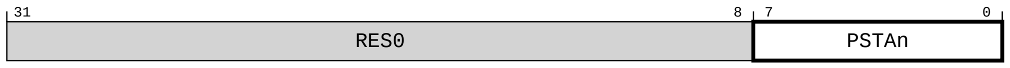
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-106: css600_jtagap_PSTA



The following table shows the bit assignments.

Table 10-111: PSTA attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	PSTAn	Read-write	Port Status. Each register field is named as PSTAn, where n = 0-7. The numerical index represents the bit position of that field in this register.

10.6.2.4 css600_jtagap Byte FIFO Registers, BFIFO1

The BFIFO Registers together enable up to four bytes to be transacted with the Response or Command FIFO, by reading or writing the appropriate register.

Attributes

Offset

`0x0D10`

Type

Read-write

Reset

`0x000000--`

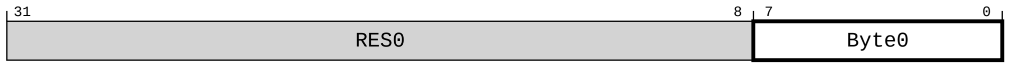
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-107: css600_jtagap_BFIFO1



The following table shows the bit assignments.

Table 10-112: BFIFO1 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[7:0]	UNKNOWN	Byte0	Read-write	First byte

10.6.2.5 css600_jtagap Byte FIFO Registers, BFIFO2

The BFIFO Registers together enable up to four bytes to be transacted with the Response or Command FIFO, by reading or writing the appropriate register.

Attributes

Offset

`0x0D14`

Type

Read-write

Reset

`0x0000----`

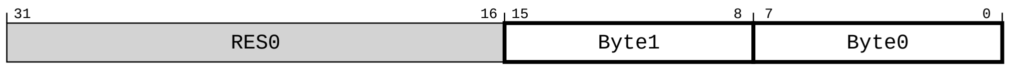
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-108: css600_jtagap_BFIFO2



The following table shows the bit assignments.

Table 10-113: BFIFO2 attributes

Bits	Reset value	Name	Type	Function
[31:16]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[15:8]	UNKNOWN	Byte1	Read-write	Second byte
[7:0]	UNKNOWN	Byte0	Read-write	First byte

10.6.2.6 css600_jtagap Byte FIFO Registers, BFIFO3

The BFIFO Registers together enable up to four bytes to be transacted with the Response or Command FIFO, by reading or writing the appropriate register.

Attributes

Offset

`0x0D18`

Type

Read-write

Reset

`0x000000--`

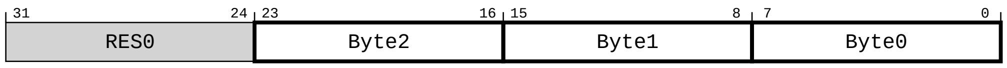
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-109: css600_jtagap_BFIFO3



The following table shows the bit assignments.

Table 10-114: BFIFO3 attributes

Bits	Reset value	Name	Type	Function
[31:24]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[23:16]	UNKNOWN	Byte2	Read-write	Third byte
[15:8]	UNKNOWN	Byte1	Read-write	Second byte
[7:0]	UNKNOWN	Byte0	Read-write	First byte

10.6.2.7 css600_jtagap Byte FIFO Registers, BFIFO4

The BFIFO Registers together enable up to four bytes to be transacted with the Response or Command FIFO, by reading or writing the appropriate register.

Attributes

Offset

`0x0D1C`

Type

Read-write

Reset

`0x--------`

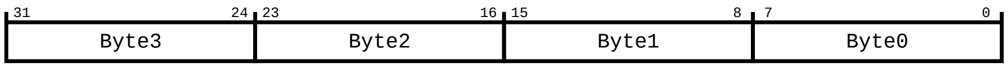
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-110: css600_jtagap_BFIFO4



The following table shows the bit assignments.

Table 10-115: BFIFO4 attributes

Bits	Reset value	Name	Type	Function
[31:24]	UNKNOWN	Byte3	Read-write	Forth byte.
[23:16]	UNKNOWN	Byte2	Read-write	Third byte.
[15:8]	UNKNOWN	Byte1	Read-write	Second byte.
[7:0]	UNKNOWN	Byte0	Read-write	First byte.

10.6.2.8 css600_jtagap Identification Register, IDR

This register provides a mechanism for the debugger to know various identity attributes of the AP.

Attributes

Offset

`0x0DFC`

Type

Read-only

Reset

`0x34760020`

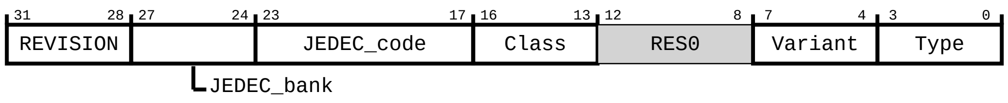
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-111: css600_jtagap_IDR



The following table shows the bit assignments.

Table 10-116: IDR attributes

Bits	Reset value	Name	Type	Function
[31:28]	`0b0011`	REVISION	Read-only	Revision. An incremental value starting at `0x0` for the first design of a component. See the Component list in Chapter 1 for information on the RTL revision of the component.
[27:24]	`0b0100`	JEDEC_bank	Read-only	The JEP106 continuation code. Returns `0x4`, indicating Arm as the designer.
[23:17]	`0x3B`	JEDEC_code	Read-only	The JEP106 identification code. Returns `0x3B`, indicating Arm as the designer.
[16:13]	`0b0000`	Class	Read-only	Returns `0x8`, indicating No defined class.
[12:8]	`0b00000`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0010`	Variant	Read-only	Returns `0x2`, indicating variation from base type specified by IDR.Type.
[3:0]	`0b0000`	Type	Read-only	Returns `0x0`, indicating that this is a JTAG Access Port.

10.6.2.9 css600_jtagap Integration Test Status register, ITSTATUS

This register indicates the Integration Test DP Abort status.

Attributes

Offset

`0x0EFC`

Type

Read-write

Reset

`0x00000000`

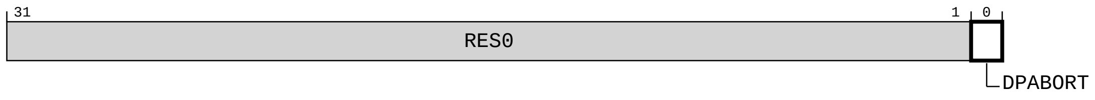
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-112: css600_jtagap_ITSTATUS



The following table shows the bit assignments.

Table 10-117: ITSTATUS attributes

Bits	Reset value	Name	Type	Function
[31:1]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[0]	`0b0`	DPABORT	Read-only	When in Integration testing mode (ITCTRL.IME=`0b1`): Behaves as a sticky bit and latches to 1 on a rising edge of dp_abort. Cleared on a read from this register. If dp_abort rises in the same cycle as a read of the ITSTATUS register is received, the read takes priority and the register is cleared. When in normal functional operation mode (ITCTRL.IME=`0b0`): Read as 0, writes ignored.

10.6.2.10 css600_jtagap Integration Mode Control Register, ITCTRL

The Integration Mode Control register is used to enable topology detection.

Attributes

Offset

`0x0F00`

Type

Read-write

Reset

`0x00000000`

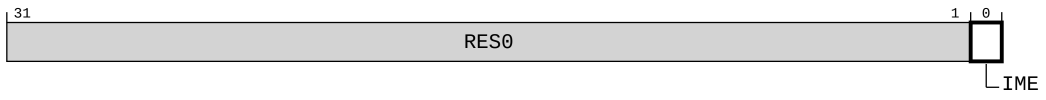
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-113: css600_jtagap_ITCTRL



The following table shows the bit assignments.

Table 10-118: ITCTRL attributes

Bits	Reset value	Name	Type	Function
[31:1]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[0]	`0b0`	IME	Read-write	Integration Mode Enable. When set, the component enters integration mode, enabling topology detection or integration testing to be performed.

10.6.2.11 css600_jtagap Claim Tag Set Register, CLAIMSET

This register forms one half of the claim tag value. On writes, this location enables individual bits to be set. On reads, it returns the number of bits that can be set.

Attributes

Offset

`0x0FA0`

Type

Read-write

Reset

`0x00000003`

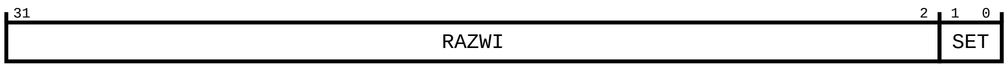
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-114: css600_jtagap_CLAIMSET



The following table shows the bit assignments.

Table 10-119: CLAIMSET attributes

Bits	Reset value	Name	Type	Function
[31:2]	`0x0`	RAZ/WI	Read-write	RAZ/WI
[1:0]	`0b11`	SET	Read-write	A bit-programmable register bank that sets the claim tag value. A read returns a logic 1 for all implemented locations.

10.6.2.12 css600_jtagap Claim Tag Clear Register, CLAIMCLR

This register forms one half of the claim tag value. On writes, this location enables individual bits to be cleared. On reads, it returns the current claim tag value.

Attributes

Offset

`0x0FA4`

Type

Read-write

Reset

`0x00000000`

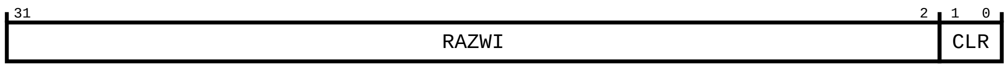
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-115: css600_jtagap_CLAIMCLR



The following table shows the bit assignments.

Table 10-120: CLAIMCLR attributes

Bits	Reset value	Name	Type	Function
[31:2]	`0x0`	RAZ/ WI	Read-write	RAZ/WI
[1:0]	`0b00`	CLR	Read-write	A bit-programmable register bank that clears the claim tag value. It is zero at reset. It is used by software agents to signal to each other ownership of the hardware. It has no direct effect on the hardware itself.

10.6.2.13 css600_jtagap Device Architecture Register, DEVARCH

Identifies the architect and architecture of a CoreSight component. The architect might differ from the designer of a component, for example Arm defines the architecture but another company designs and implements the component.

Attributes

Offset

`0x0FBC`

Type

Read-only

Reset

`0x47700A27`

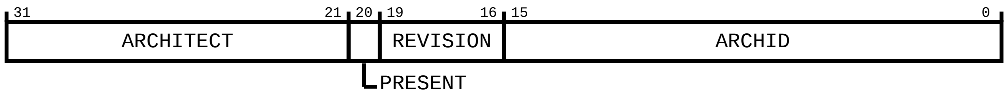
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-116: css600_jtagap_DEVARCH



The following table shows the bit assignments.

Table 10-121: DEVARCH attributes

Bits	Reset value	Name	Type	Function
[31:21]	`0x23B`	ARCHITECT	Read-only	Returns 0x23B, denoting Arm as architect of the component
[20]	`0b1`	PRESENT	Read-only	Returns 1, indicating that the DEVARCH register is present
[19:16]	`0b0000`	REVISION	Read-only	Architecture revision. Returns the revision of the architecture that the ARCHID field specifies.
[15:0]	`0xA27`	ARCHID	Read-only	Architecture ID. Returns 0x0A27, identifying APv2 JTAG-AP architecture v0.

10.6.2.14 css600_jtagap Device Type Identifier Register, DEVTYPE

A debugger can use this register to get information about a component that has an unrecognized Part number.

Attributes

Offset

`0x0FCC`

Type

Read-only

Reset

`0x00000000`

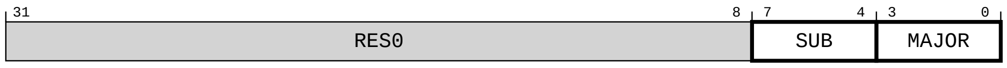
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-117: css600_jtagap_DEVTYPE



The following table shows the bit assignments.

Table 10-122: DEVTYPE attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0000`	SUB	Read-only	Minor classification. Returns `0x0`, Other/undefined.
[3:0]	`0b0000`	MAJOR	Read-only	Major classification. Returns `0x0`, Miscellaneous.

10.6.2.15 css600_jtagap Peripheral Identification Register 4, PIDR4

The PIDR4 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FD0`

Type

Read-only

Reset

`0x00000004`

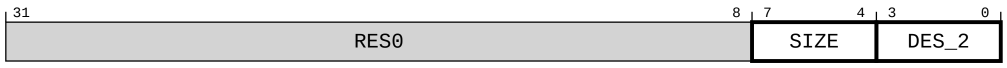
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-118: css600_jtagap_PIDR4



The following table shows the bit assignments.

Table 10-123: PIDR4 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0000`	SIZE	Read-only	Indicates the memory size that is used by this component. Returns 0 indicating that the component uses an **UNKNOWN** number of 4KB blocks. Using the SIZE field to indicate the size of the component is deprecated.
[3:0]	`0b0100`	DES_2	Read-only	JEP106 continuation code. Together, with PIDR2.DES_1 and PIDR1.DES_0, they indicate the designer of the component and not the implementer, except where the two are the same.

10.6.2.16 css600_jtagap Peripheral Identification Register 5, PIDR5

The PIDR5 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FD4`

Type

Read-only

Reset

`0x00000000`

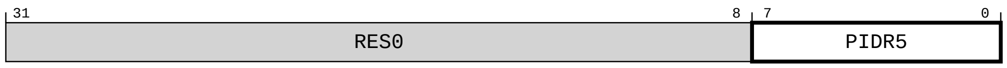
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-119: css600_jtagap_PIDR5



The following table shows the bit assignments.

Table 10-124: PIDR5 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	PIDR5	Read-only	Reserved.

10.6.2.17 css600_jtagap Peripheral Identification Register 6, PIDR6

The PIDR6 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FD8`

Type

Read-only

Reset

`0x00000000`

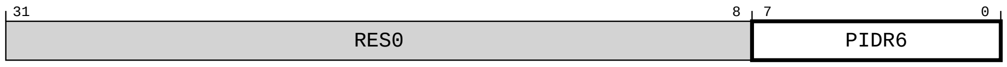
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-120: css600_jtagap_PIDR6



The following table shows the bit assignments.

Table 10-125: PIDR6 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	PIDR6	Read-only	Reserved.

10.6.2.18 css600_jtagap Peripheral Identification Register 7, PIDR7

The PIDR7 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FDC`

Type

Read-only

Reset

`0x00000000`

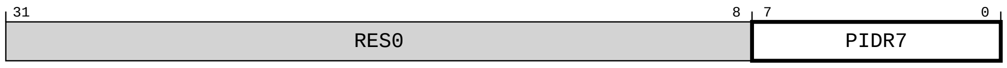
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-121: css600_jtagap_PIDR7



The following table shows the bit assignments.

Table 10-126: PIDR7 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	PIDR7	Read-only	Reserved.

10.6.2.19 css600_jtagap Peripheral Identification Register 0, PIDR0

The PIDR0 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FE0`

Type

Read-only

Reset

`0x000000E6`

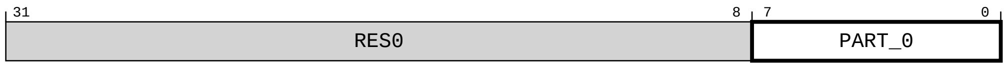
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-122: css600_jtagap_PIDR0



The following table shows the bit assignments.

Table 10-127: PIDR0 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	0xE6	PART_0	Read-only	Part number, bits[7:0]. Taken together with PIDR1.PART_1 it indicates the component. The Part Number is selected by the designer of the component.

10.6.2.20 css600_jtagap Peripheral Identification Register 1, PIDR1

The PIDR1 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FE4`

Type

Read-only

Reset

`0x000000B9`

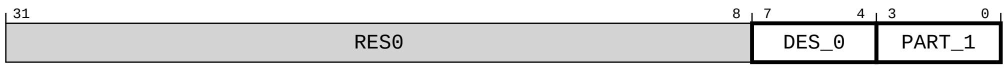
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-123: css600_jtagap_PIDR1



The following table shows the bit assignments.

Table 10-128: PIDR1 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b1011`	DES_0	Read-only	JEP106 identification code, bits[3:0]. Together, with PIDR4.DES_2 and PIDR2.DES_1, they indicate the designer of the component and not the implementer, except where the two are the same.
[3:0]	`0b1001`	PART_1	Read-only	Part number, bits[11:8]. Taken together with PIDR0.PART_0 it indicates the component. The Part Number is selected by the designer of the component.

10.6.2.21 css600_jtagap Peripheral Identification Register 2, PIDR2

The PIDR2 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FE8`

Type

Read-only

Reset

`0x0000003B`

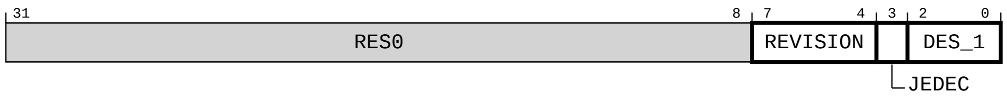
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-124: css600_jtagap_PIDR2



The following table shows the bit assignments.

Table 10-129: PIDR2 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0011`	REVISION	Read-only	Revision. It is an incremental value starting at `0x0` for the first design of a component. See the Component list in Chapter 1 for information on the RTL revision of the component.
[3]	`0b1`	JEDEC	Read-only	1 - Always set. Indicates that a JEDEC assigned value is used.
[2:0]	`0b011`	DES_1	Read-only	JEP106 identification code, bits[6:4]. Together, with PIDR4.DES_2 and PIDR1.DES_0, they indicate the designer of the component and not the implementer, except where the two are the same.

10.6.2.22 css600_jtagap Peripheral Identification Register 3, PIDR3

The PIDR3 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FEC`

Type

Read-only

Reset

`0x00000000`

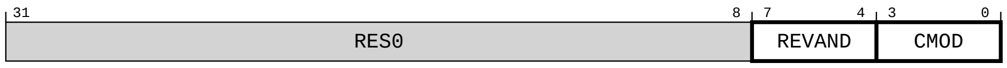
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-125: css600_jtagap_PIDR3



The following table shows the bit assignments.

Table 10-130: PIDR3 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0000`	REVAND	Read-only	This field indicates minor errata fixes specific to this design, for example metal fixes after implementation. In most cases this field is `0x0`.
[3:0]	`0b0000`	CMOD	Read-only	Customer Modified. Where the component is reusable IP, this value indicates if the customer has modified the behavior of the component. In most cases this field is `0x0`.

10.6.2.23 css600_jtagap Component Identification Register 0, CIDR0

The CIDR0 register is part of the set of component identification registers.

Attributes

Offset

`0x0FF0`

Type

Read-only

Reset

`0x0000000D`

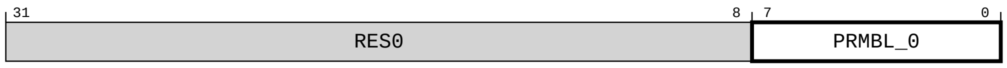
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-126: css600_jtagap_CIDR0



The following table shows the bit assignments.

Table 10-131: CIDR0 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0xD`	PRMBL_0	Read-only	Preamble. Returns `0x0D`.

10.6.2.24 css600_jtagap Component Identification Register 1, CIDR1

The CIDR1 register is part of the set of component identification registers.

Attributes

Offset

`0x0FF4`

Type

Read-only

Reset

`0x00000090`

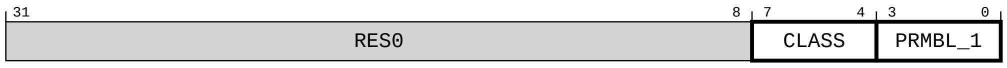
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-127: css600_jtagap_CIDR1



The following table shows the bit assignments.

Table 10-132: CIDR1 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b1001`	CLASS	Read-only	Component class. Returns `0x9`, indicating this is a CoreSight component.
[3:0]	`0b0000`	PRMBL_1	Read-only	Preamble. Returns `0x0`.

10.6.2.25 css600_jtagap Component Identification Register 2, CIDR2

The CIDR2 register is part of the set of component identification registers.

Attributes

Offset

`0x0FF8`

Type

Read-only

Reset

`0x00000005`

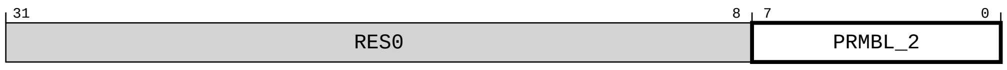
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-128: css600_jtagap_CIDR2



The following table shows the bit assignments.

Table 10-133: CIDR2 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x5`	PRMBL_2	Read-only	Preamble. Returns `0x05`.

10.6.2.26 css600_jtagap Component Identification Register 3, CIDR3

The CIDR3 register is part of the set of component identification registers.

Attributes

Offset
`0x0FFC`

Type
Read-only

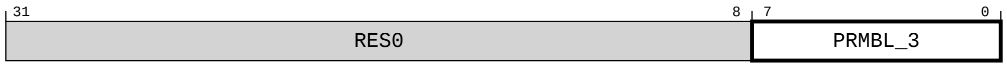
Reset
`0x000000B1`

Width
32

Bit descriptions

The following image shows the bit assignments.

Figure 10-129: css600_jtagap_CIDR3



The following table shows the bit assignments.

Table 10-134: CIDR3 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0xB1`	PRMBL_3	Read-only	Preamble. Returns `0xB1`.

10.7 `css600_apbrom` introduction

This section describes the programmers model of the `css600_apbrom`.

10.7.1 css600_apbrom register summary

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

Note

A reset value containing one or more '-' means that this register contains **UNKNOWN** or **IMPLEMENTATION-DEFINED** values. See the relevant register description for more information.

Locations that are not listed in the table are Reserved.

Table 10-135: css600_apbrom register summary

Offset	Name	Type	Reset	Width	Description
`0x0000`	ROMEntry0	RO	`0x--------`	32	css600_apbrom ROM Entries register 0, ROMEntry0
`0x0004`	ROMEntry1	RO	`0x--------`	32	css600_apbrom ROM Entries register 1, ROMEntry1
`0x0008`	ROMEntry2	RO	`0x--------`	32	css600_apbrom ROM Entries register 2, ROMEntry2
...	...	...	...	...	...
`0x07FC`	ROMEntry511	RO	`0x--------`	32	css600_apbrom ROM Entries register 511, ROMEntry511
`0x0FB8`	AUTHSTATUS	RO	`0x000000--`	32	css600_apbrom Authentication Status Register, AUTHSTATUS
`0x0FBC`	DEVARCH	RO	`0x47700AF7`	32	css600_apbrom Device Architecture Register, DEVARCH
`0x0FC8`	DEVID	RO	`0x000000-0`	32	css600_apbrom Device Configuration Register, DEVID
`0x0FD0`	PIDR4	RO	`0x0000000-`	32	css600_apbrom Peripheral Identification Register 4, PIDR4
`0x0FD4`	PIDR5	RO	`0x00000000`	32	css600_apbrom Peripheral Identification Register 5, PIDR5
`0x0FD8`	PIDR6	RO	`0x00000000`	32	css600_apbrom Peripheral Identification Register 6, PIDR6
`0x0FDC`	PIDR7	RO	`0x00000000`	32	css600_apbrom Peripheral Identification Register 7, PIDR7
`0x0FE0`	PIDR0	RO	`0x000000--`	32	css600_apbrom Peripheral Identification Register 0, PIDR0
`0x0FE4`	PIDR1	RO	`0x000000--`	32	css600_apbrom Peripheral Identification Register 1, PIDR1
`0x0FE8`	PIDR2	RO	`0x000000--`	32	css600_apbrom Peripheral Identification Register 2, PIDR2
`0x0FEC`	PIDR3	RO	`0x000000-0`	32	css600_apbrom Peripheral Identification Register 3, PIDR3
`0x0FF0`	CIDR0	RO	`0x0000000D`	32	css600_apbrom Component Identification Register 0, CIDR0
`0x0FF4`	CIDR1	RO	`0x00000090`	32	css600_apbrom Component Identification Register 1, CIDR1
`0x0FF8`	CIDR2	RO	`0x00000005`	32	css600_apbrom Component Identification Register 2, CIDR2
`0x0FFC`	CIDR3	RO	`0x000000B1`	32	css600_apbrom Component Identification Register 3, CIDR3

10.7.2 Register descriptions

This section describes the `css600_apbrom` registers.

css600_apbrom register summary provides cross references to individual registers.

Note

The ROM table has a configuration parameter `TIE_OFF_PRESENT`, which, if set to 1, allows for the selective removal of any entry using the entry_present input bus. The entry_present bus contains one bit per ROM table entry. If a given entry_present[n] bit is tied HIGH, then the value in ROMEntry<n>.PRESENT[1:0] is taken directly from the value in the `ROM_ENTRY<n>` parameter for that entry. If the entry_present[n] input is tied LOW, then a value of `0x3` in ROMEntry<n>.PRESENT[1:0] is modified to read `0x2` to indicate that the value is not present and is not the last entry.

10.7.2.1 css600_apbrom ROM Entries register 0, ROMEntry0

The ROMEntry register contains a descripter of a CoreSight component in the system. All ROM table entries conform to the same format.

Attributes

Offset

`0x0000`

Type

Read-only

Reset

`0x--------`

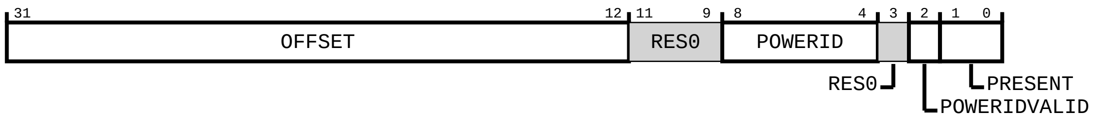
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-130: css600_apbrom_ROMEntry0



The following table shows the bit assignments.

Table 10-136: ROMEntry0 attributes

Bits	Reset value	Name	Type	Function
[31:12]	IMPLEMENTATION DEFINED	OFFSET	Read-only	The component address, relative to the base address of this ROM table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). If a component occupies more than a single 4KB block, OFFSET points to the 4KB block which contains the Peripheral ID and Component ID registers for the component. Negative values of OFFSET are permitted, using two's complement.
[11:9]	IMPLEMENTATION DEFINED	RES0	Read-only	Reserved bit or field with SBZP behavior
[8:4]	IMPLEMENTATION DEFINED	POWERID	Read-only	Indicates the power domain ID of the component. Only valid if bit 2 is set. If bit 2 is clear then this field has a value of 0. Possible values are 0 to 31, representing the 32 DBGPWRUPREQ/ACK interface pins of the component.
[3]	IMPLEMENTATION DEFINED	RES0	Read-only	Reserved bit or field with SBZP behavior
[2]	IMPLEMENTATION DEFINED	POWERIDVALID	Read-only	Indicates whether there is a power domain ID specified in the ROM table entry: **0** POWERID field of this register is not valid **1** POWERID field of this register is valid

Bits	Reset value	Name	Type	Function
[1:0]	IMPLEMENTATION DEFINED	PRESENT	Read-only	Indicates whether the ROM table entry is present: `0x0` ROM table entry not present. This is the last entry. `0x1` Reserved `0x2` ROM table entry not present. This is not the last entry. `0x3` ROM table entry present

10.7.2.2 css600_apbrom ROM Entries register 1, ROMEntry1

The ROMEntry register contains a descripter of a CoreSight component in the system. All ROM table entries conform to the same format.

Attributes

Offset

`0x0004`

Type

Read-only

Reset

`0x--------`

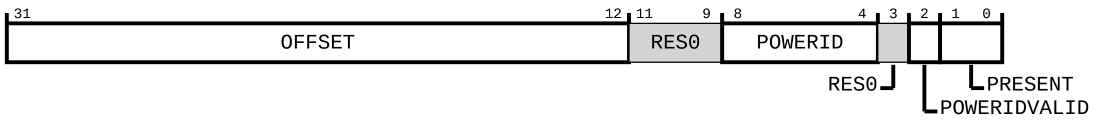
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-131: css600_apbrom_ROMEntry1



The following table shows the bit assignments.

Table 10-137: ROMEntry1 attributes

Bits	Reset value	Name	Type	Function
[31:12]	IMPLEMENTATION DEFINED	OFFSET	Read-only	The component address, relative to the base address of this ROM table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). If a component occupies more than a single 4KB block, OFFSET points to the 4KB block which contains the Peripheral ID and Component ID registers for the component. Negative values of OFFSET are permitted, using two's complement.
[11:9]	IMPLEMENTATION DEFINED	RES0	Read-only	Reserved bit or field with SBZP behavior

Bits	Reset value	Name	Type	Function
[8:4]	IMPLEMENTATION DEFINED	POWERID	Read-only	Indicates the power domain ID of the component. Only valid if bit 2 is set. If bit 2 is clear then this field has a value of 0. Possible values are 0 to 31, representing the 32 DBGPWRUPREQ/ACK interface pins of the component.
[3]	IMPLEMENTATION DEFINED	RES0	Read-only	Reserved bit or field with SBZP behavior
[2]	IMPLEMENTATION DEFINED	POWERIDVALID	Read-only	Indicates whether there is a power domain ID specified in the ROM table entry: **0** POWERID field of this register is not valid **1** POWERID field of this register is valid
[1:0]	IMPLEMENTATION DEFINED	PRESENT	Read-only	Indicates whether the ROM table entry is present: **`0x0`** ROM table entry not present. This is the last entry. **`0x1`** Reserved **`0x2`** ROM table entry not present. This is not the last entry. **`0x3`** ROM table entry present

10.7.2.3 css600_apbrom ROM Entries register 2, ROMEntry2

The ROMEntry register contains a descripter of a CoreSight component in the system. All ROM table entries conform to the same format.

Attributes

Offset

`0x0008`

Type

Read-only

Reset

`0x--------`

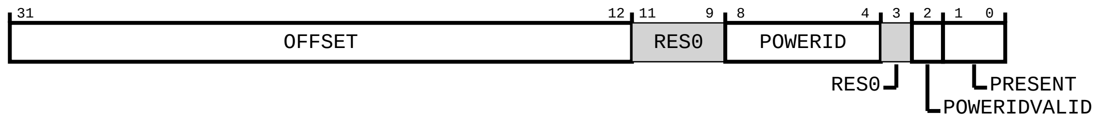
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-132: css600_apbrom_ROMEntry2



The following table shows the bit assignments.

Table 10-138: ROMEntry2 attributes

Bits	Reset value	Name	Type	Function
[31:12]	IMPLEMENTATION DEFINED	OFFSET	Read-only	The component address, relative to the base address of this ROM table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). If a component occupies more than a single 4KB block, OFFSET points to the 4KB block which contains the Peripheral ID and Component ID registers for the component. Negative values of OFFSET are permitted, using two's complement.
[11:9]	IMPLEMENTATION DEFINED	RES0	Read-only	Reserved bit or field with SBZP behavior
[8:4]	IMPLEMENTATION DEFINED	POWERID	Read-only	Indicates the power domain ID of the component. Only valid if bit 2 is set. If bit 2 is clear then this field has a value of 0. Possible values are 0 to 31, representing the 32 DBGPWRUPREQ/ACK interface pins of the component.
[3]	IMPLEMENTATION DEFINED	RES0	Read-only	Reserved bit or field with SBZP behavior
[2]	IMPLEMENTATION DEFINED	POWERIDVALID	Read-only	Indicates whether there is a power domain ID specified in the ROM table entry: **0** POWERID field of this register is not valid **1** POWERID field of this register is valid
[1:0]	IMPLEMENTATION DEFINED	PRESENT	Read-only	Indicates whether the ROM table entry is present: **0x0** ROM table entry not present. This is the last entry. **0x1** Reserved **0x2** ROM table entry not present. This is not the last entry. **0x3** ROM table entry present

10.7.2.4 css600_apbrom ROM Entries register 511, ROMEntry511

The ROMEntry register contains a descripter of a CoreSight component in the system. All ROM table entries conform to the same format.

Attributes

Offset

`0x07FC`

Type

Read-only

Reset

`0x--------`

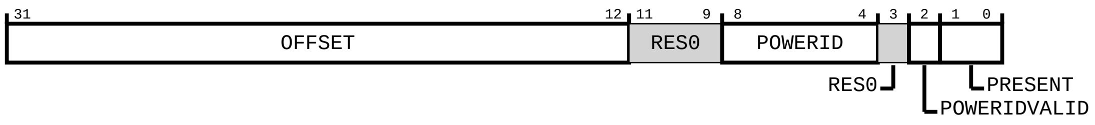
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-133: css600_apbrom_ROMEntry511



The following table shows the bit assignments.

Table 10-139: ROMEntry511 attributes

Bits	Reset value	Name	Type	Function
[31:12]	**IMPLEMENTATION DEFINED**	OFFSET	Read-only	The component address, relative to the base address of this ROM table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). If a component occupies more than a single 4KB block, OFFSET points to the 4KB block which contains the Peripheral ID and Component ID registers for the component. Negative values of OFFSET are permitted, using two's complement.
[11:9]	**IMPLEMENTATION DEFINED**	RES0	Read-only	Reserved bit or field with SBZP behavior
[8:4]	**IMPLEMENTATION DEFINED**	POWERID	Read-only	Indicates the power domain ID of the component. Only valid if bit 2 is set. If bit 2 is clear then this field has a value of 0. Possible values are 0 to 31, representing the 32 DBGPWRUPREQ/ACK interface pins of the component.
[3]	**IMPLEMENTATION DEFINED**	RES0	Read-only	Reserved bit or field with SBZP behavior
[2]	**IMPLEMENTATION DEFINED**	POWERIDVALID	Read-only	Indicates whether there is a power domain ID specified in the ROM table entry: **0** POWERID field of this register is not valid **1** POWERID field of this register is valid
[1:0]	**IMPLEMENTATION DEFINED**	PRESENT	Read-only	Indicates whether the ROM table entry is present: **0x0** ROM table entry not present. This is the last entry. **0x1** Reserved **0x2** ROM table entry not present. This is not the last entry. **0x3** ROM table entry present

10.7.2.5 css600_apbrom Authentication Status Register, AUTHSTATUS

Reports the current status of the authentication control signals.

Attributes

Offset

`0x0FB8`

Type

Read-only

Reset

`0x000000--`

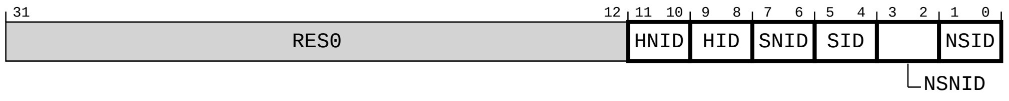
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-134: css600_apbrom_AUTHSTATUS



The following table shows the bit assignments.

Table 10-140: AUTHSTATUS attributes

Bits	Reset value	Name	Type	Function
[31:12]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[11:10]	`0b00`	HNID	Read-only	Hypervisor non-invasive debug. **0x0** Functionality not implemented or controlled elsewhere. **0x1** Reserved. **0x2** Functionality disabled. **0x3** Functionality enabled.
[9:8]	`0b00`	HID	Read-only	Hypervisor invasive debug. **0x0** Functionality not implemented or controlled elsewhere. **0x1** Reserved. **0x2** Functionality disabled. **0x3** Functionality enabled.
[7:6]	UNKNOWN	SNID	Read-only	Secure non-invasive debug. **0x0** Functionality not implemented or controlled elsewhere. **0x1** Reserved. **0x2** Functionality disabled. **0x3** Functionality enabled.
[5:4]	UNKNOWN	SID	Read-only	Secure invasive debug. **0x0** Functionality not implemented or controlled elsewhere. **0x1** Reserved. **0x2** Functionality disabled. **0x3** Functionality enabled.
[3:2]	UNKNOWN	NSNID	Read-only	Non-secure non-invasive debug. **0x0** Functionality not implemented or controlled elsewhere. **0x1** Reserved. **0x2** Functionality disabled. **0x3** Functionality enabled.

Bits	Reset value	Name	Type	Function
[1:0]	UNKNOWN	NSID	Read-only	Non-secure invasive debug. **0x0** Functionality not implemented or controlled elsewhere. **0x1** Reserved. **0x2** Functionality disabled. **0x3** Functionality enabled.

10.7.2.6 css600_apbrom Device Architecture Register, DEVARCH

Identifies the architect and architecture of a CoreSight component. The architect might differ from the designer of a component, for example Arm defines the architecture but another company designs and implements the component.

Attributes

Offset

`0x0FBC`

Type

Read-only

Reset

`0x47700AF7`

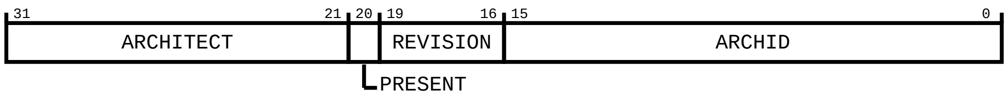
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-135: css600_apbrom_DEVARCH



The following table shows the bit assignments.

Table 10-141: DEVARCH attributes

Bits	Reset value	Name	Type	Function
[31:21]	`0x23B`	ARCHITECT	Read-only	Returns 0x23B, denoting Arm as architect of the component
[20]	`0b1`	PRESENT	Read-only	Returns 1, indicating that the DEVARCH register is present
[19:16]	`0b0000`	REVISION	Read-only	Architecture revision. Returns the revision of the architecture that the ARCHID field specifies.

Bits	Reset value	Name	Type	Function
[15:0]	`0xAF7`	ARCHID	Read-only	Architecture ID. Returns 0x0AF7, identifying ROM Table Architecture v0.

10.7.2.7 css600_apbrom Device Configuration Register, DEVID

This register is **IMPLEMENTATION DEFINED** for each Part Number and Designer. The register indicates the capabilities of the component.

Attributes

Offset

`0x0FC8`

Type

Read-only

Reset

`0x000000-0`

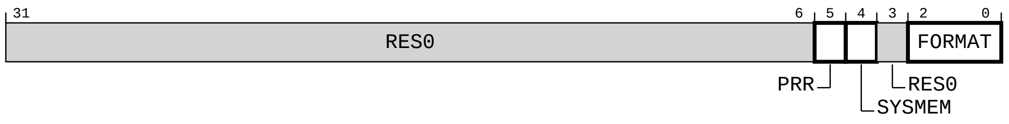
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-136: css600_apbrom_DEVID



The following table shows the bit assignments.

Table 10-142: DEVID attributes

Bits	Reset value	Name	Type	Function
[31:6]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[5]	`0b0`	PRR	Read-only	Indicates that power request functionality is included. Set by the GPR_PRESENT parameter. **0** GPR is not included (css600_apbrom) **1** GPR is included (css600_apbrom_gpr)

Bits	Reset value	Name	Type	Function
[4]	**IMPLEMENTATION DEFINED**	SYSMEM	Read-only	Indicates whether system memory is present on the bus. Set by the SYSMEM parameter. **0** System memory is not present and the bus is a dedicated debug bus **1** Indicates that there is system memory on the bus
[3]	`0b0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[2:0]	`0b000`	FORMAT	Read-only	Indicates that this is a 32-bit ROM table.

10.7.2.8 css600_apbrom Peripheral Identification Register 4, PIDR4

The PIDR4 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FD0`

Type

Read-only

Reset

`0x0000000-`

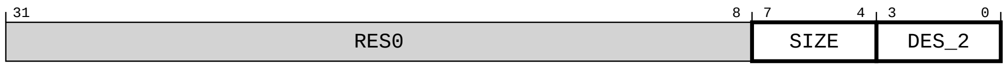
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-137: css600_apbrom_PIDR4



The following table shows the bit assignments.

Table 10-143: PIDR4 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0000`	SIZE	Read-only	Indicates the memory size that is used by this component. Returns 0 indicating that the component uses an **UNKNOWN** number of 4KB blocks. Using the SIZE field to indicate the size of the component is deprecated.
[3:0]	**IMPLEMENTATION DEFINED**	DES_2	Read-only	JEP106 continuation code. Together, with PIDR2.DES_1 and PIDR1.DES_0, they indicate the designer of the component and not the implementer, except where the two are the same.

10.7.2.9 css600_apbrom Peripheral Identification Register 5, PIDR5

The PIDR5 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FD4`

Type

Read-only

Reset

`0x00000000`

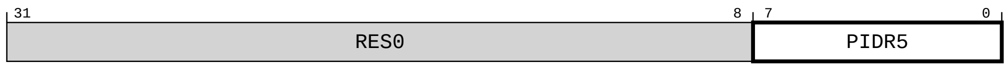
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-138: css600_apbrom_PIDR5



The following table shows the bit assignments.

Table 10-144: PIDR5 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	PIDR5	Read-only	Reserved.

10.7.2.10 css600_apbrom Peripheral Identification Register 6, PIDR6

The PIDR6 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FD8`

Type

Read-only

Reset

`0x00000000`

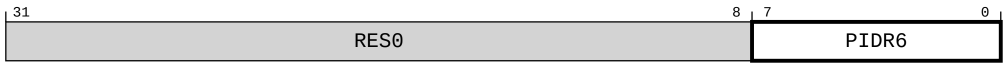
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-139: css600_apbrom_PIDR6



The following table shows the bit assignments.

Table 10-145: PIDR6 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	PIDR6	Read-only	Reserved.

10.7.2.11 css600_apbrom Peripheral Identification Register 7, PIDR7

The PIDR7 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FDC`

Type

Read-only

Reset

`0x00000000`

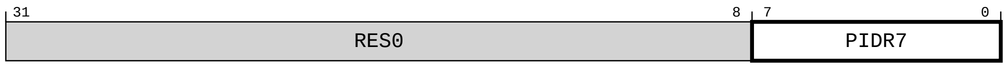
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-140: css600_apbrom_PIDR7



The following table shows the bit assignments.

Table 10-146: PIDR7 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	PIDR7	Read-only	Reserved.

10.7.2.12 css600_apbrom Peripheral Identification Register 0, PIDR0

The PIDR0 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FE0`

Type

Read-only

Reset

`0x000000--`

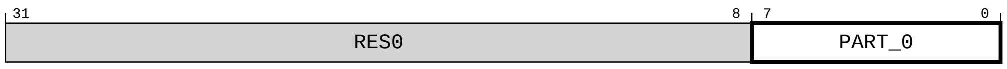
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-141: css600_apbrom_PIDR0



The following table shows the bit assignments.

Table 10-147: PIDR0 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	IMPLEMENTATION DEFINED	PART_0	Read-only	Part number, bits[7:0]. Set by the configuration inputs part_number[7:0]

10.7.2.13 css600_apbrom Peripheral Identification Register 1, PIDR1

The PIDR1 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FE4`

Type

Read-only

Reset

`0x000000--`

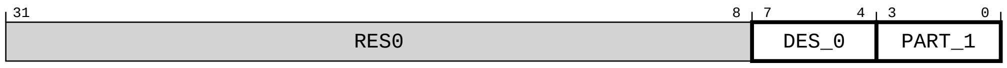
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-142: css600_apbrom_PIDR1



The following table shows the bit assignments.

Table 10-148: PIDR1 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	IMPLEMENTATION DEFINED	DES_0	Read-only	JEP106 identification code, bits[3:0]. Set by the configuration inputs jep106_id[3:0]. Together, with PIDR4.DES_2 and PIDR2.DES_1, they indicate the designer of the component and not the implementer, except where the two are the same.
[3:0]	IMPLEMENTATION DEFINED	PART_1	Read-only	Part number, bits[11:8]. Set by the configuration inputs part_number[11:8].

10.7.2.14 css600_apbrom Peripheral Identification Register 2, PIDR2

The PIDR2 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FE8`

Type

Read-only

Reset

`0x000000--`

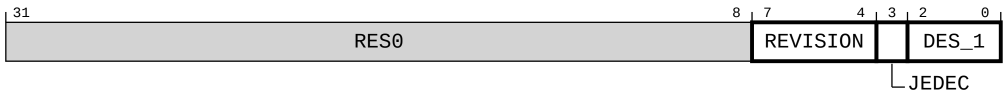
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-143: css600_apbrom_PIDR2



The following table shows the bit assignments.

Table 10-149: PIDR2 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	IMPLEMENTATION DEFINED	REVISION	Read-only	Revision. Set by the configuration inputs revision[3:0].
[3]	`0b1`	JEDEC	Read-only	1 - Always set. Indicates that a JEDEC assigned value is used.
[2:0]	IMPLEMENTATION DEFINED	DES_1	Read-only	JEP106 identification code, bits[6:4]. Set by the configuration inputs jep106_id[6:4]. Together, with PIDR4.DES_2 and PIDR1.DES_0, they indicate the designer of the component and not the implementer, except where the two are the same.

10.7.2.15 css600_apbrom Peripheral Identification Register 3, PIDR3

The PIDR3 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FEC`

Type

Read-only

Reset

`0x000000-0`

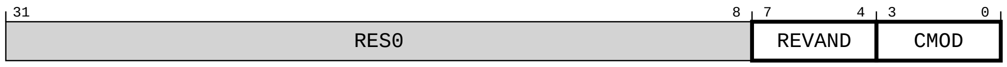
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-144: css600_apbrom_PIDR3



The following table shows the bit assignments.

Table 10-150: PIDR3 attributes

Bits	Reset value	Name	Type	Function
[31:8]	0x0	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	0b0000	REVAND	Read-only	This field indicates minor errata fixes specific to this design, for example metal fixes after implementation. In most cases this field is 0x0.
[3:0]	0b0000	CMOD	Read-only	Customer Modified. Where the component is reusable IP, this value indicates if the customer has modified the behavior of the component. In most cases this field is 0x0.

10.7.2.16 css600_apbrom Component Identification Register 0, CIDR0

The CIDR0 register is part of the set of component identification registers.

Attributes

Offset

0x0FF0

Type

Read-only

Reset

0x0000000D

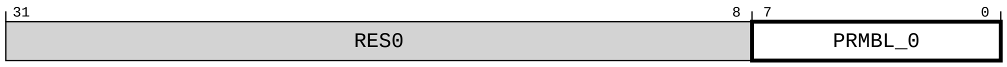
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-145: css600_apbrom_CIDR0



The following table shows the bit assignments.

Table 10-151: CIDR0 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0xD`	PRMBL_0	Read-only	Preamble. Returns `0x0D`.

10.7.2.17 css600_apbrom Component Identification Register 1, CIDR1

The CIDR1 register is part of the set of component identification registers.

Attributes

Offset

`0x0FF4`

Type

Read-only

Reset

`0x00000090`

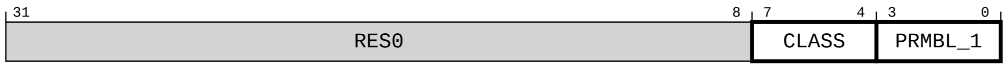
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-146: css600_apbrom_CIDR1



The following table shows the bit assignments.

Table 10-152: CIDR1 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b1001`	CLASS	Read-only	Component class. Returns `0x9`, indicating this is a CoreSight component.
[3:0]	`0b0000`	PRMBL_1	Read-only	Preamble. Returns `0x0`.

10.7.2.18 css600_apbrom Component Identification Register 2, CIDR2

The CIDR2 register is part of the set of component identification registers.

Attributes

Offset

`0x0FF8`

Type

Read-only

Reset

`0x00000005`

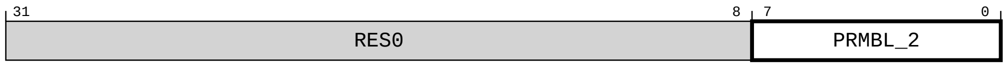
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-147: css600_apbrom_CIDR2



The following table shows the bit assignments.

Table 10-153: CIDR2 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x5`	PRMBL_2	Read-only	Preamble. Returns `0x05`.

10.7.2.19 css600_apbrom Component Identification Register 3, CIDR3

The CIDR3 register is part of the set of component identification registers.

Attributes

Offset

`0x0FFC`

Type

Read-only

Reset

`0x000000B1`

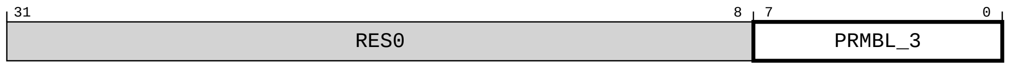
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-148: css600_apbrom_CIDR3



The following table shows the bit assignments.

Table 10-154: CIDR3 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0xB1`	PRMBL_3	Read-only	Preamble. Returns `0xB1`.

10.8 `css600_apbrom_gpr` introduction

This section describes the programmers model of the `css600_apbrom_gpr`.

10.8.1 css600_apbrom_gpr register summary

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

Note

A reset value containing one or more '-' means that this register contains unknown or implementation-defined values. See the relevant register description for more information.

Locations that are not listed in the table are Reserved.

Table 10-155: css600_apbrom_gpr register summary

Offset	Name	Type	Reset	Width	Description
`0x0000`	ROMEntry0	RO	`0x--------`	32	css600_apbrom_gpr ROM Entries register 0, ROMEntry0
`0x0004`	ROMEntry1	RO	`0x--------`	32	css600_apbrom_gpr ROM Entries register 1, ROMEntry1
`0x0008`	ROMEntry2	RO	`0x--------`	32	css600_apbrom_gpr ROM Entries register 2, ROMEntry2
…	…	…	…	…	…
`0x07FC`	ROMEntry511	RO	`0x--------`	32	css600_apbrom_gpr ROM Entries register 511, ROMEntry511
`0x0A00`	DBGPCR0	RW	`0x00000001`	32	css600_apbrom_gpr Debug Power Control Register 0, DBGPCR0
`0x0A04`	DBGPCR1	RW	`0x0000000-`	32	css600_apbrom_gpr Debug Power Control Register 1, DBGPCR1

Offset	Name	Type	Reset	Width	Description
0x0A08	DBGPCR2	RW	0x0000000-	32	css600_apbrom_gpr Debug Power Control Register 2, DBGPCR2
...	...	...	...	...	...
0x0A7C	DBGPCR31	RW	0x0000000-	32	css600_apbrom_gpr Debug Power Control Register 31, DBGPCR31
0x0A80	DBGPSR0	RO	0x0000000-	32	css600_apbrom_gpr Debug Power Status Register 0, DBGPSR0
0x0A84	DBGPSR1	RO	0x0000000-	32	css600_apbrom_gpr Debug Power Status Register 1, DBGPSR1
0x0A88	DBGPSR2	RO	0x0000000-	32	css600_apbrom_gpr Debug Power Status Register 2, DBGPSR2
...	...	...	...	...	...
0x0AFC	DBGPSR31	RO	0x0000000-	32	css600_apbrom_gpr Debug Power Status Register 31, DBGPSR31
0x0B00	SYSPCR0	RW	0x00000001	32	css600_apbrom_gpr System Power Control Register 0, SYSPCR0
0x0B04	SYSPCR1	RW	0x0000000-	32	css600_apbrom_gpr System Power Control Register 1, SYSPCR1
0x0B08	SYSPCR2	RW	0x0000000-	32	css600_apbrom_gpr System Power Control Register 2, SYSPCR2
...	...	...	...	...	...
0x0B7C	SYSPCR31	RW	0x0000000-	32	css600_apbrom_gpr System Power Control Register 31, SYSPCR31
0x0B80	SYSPSR0	RO	0x0000000-	32	css600_apbrom_gpr System Power Status Register 0, SYSPSR0
0x0B84	SYSPSR1	RO	0x0000000-	32	css600_apbrom_gpr System Power Status Register 1, SYSPSR1
0x0B88	SYSPSR2	RO	0x0000000-	32	css600_apbrom_gpr System Power Status Register 2, SYSPSR2
...	...	...	...	...	...
0x0BFC	SYSPSR31	RO	0x0000000-	32	css600_apbrom_gpr System Power Status Register 31, SYSPSR31
0x0C00	PRIDR0	RO	0x00000031	32	css600_apbrom_gpr Power Request ID Register, PRIDR0
0x0C10	DBGRSTRR	RW	0x00000000	32	css600_apbrom_gpr Debug Reset Request Register, DBGRSTRR
0x0C14	DBGRSTAR	RO	0x00000000	32	css600_apbrom_gpr Debug Reset Acknowledge Register, DBGRSTAR
0x0C18	SYSRSTRR	RW	0x00000000	32	css600_apbrom_gpr System Reset Request Register, SYSRSTRR
0x0C1C	SYSRSTAR	RO	0x00000000	32	css600_apbrom_gpr System Reset Acknowledge Register, SYSRSTAR
0x0FB8	AUTHSTATUS	RO	0x000000--	32	css600_apbrom_gpr Authentication Status Register, AUTHSTATUS
0x0FBC	DEVARCH	RO	0x47700AF7	32	css600_apbrom_gpr Device Architecture Register, DEVARCH
0x0FC8	DEVID	RO	0x000000-0	32	css600_apbrom_gpr Device Configuration Register, DEVID
0x0FD0	PIDR4	RO	0x0000000-	32	css600_apbrom_gpr Peripheral Identification Register 4, PIDR4
0x0FD4	PIDR5	RO	0x00000000	32	css600_apbrom_gpr Peripheral Identification Register 5, PIDR5
0x0FD8	PIDR6	RO	0x00000000	32	css600_apbrom_gpr Peripheral Identification Register 6, PIDR6
0x0FDC	PIDR7	RO	0x00000000	32	css600_apbrom_gpr Peripheral Identification Register 7, PIDR7
0x0FE0	PIDR0	RO	0x000000--	32	css600_apbrom_gpr Peripheral Identification Register 0, PIDR0
0x0FE4	PIDR1	RO	0x000000--	32	css600_apbrom_gpr Peripheral Identification Register 1, PIDR1
0x0FE8	PIDR2	RO	0x000000--	32	css600_apbrom_gpr Peripheral Identification Register 2, PIDR2
0x0FEC	PIDR3	RO	0x000000-0	32	css600_apbrom_gpr Peripheral Identification Register 3, PIDR3
0x0FF0	CIDR0	RO	0x0000000D	32	css600_apbrom_gpr Component Identification Register 0, CIDR0
0x0FF4	CIDR1	RO	0x00000090	32	css600_apbrom_gpr Component Identification Register 1, CIDR1
0x0FF8	CIDR2	RO	0x00000005	32	css600_apbrom_gpr Component Identification Register 2, CIDR2
0x0FFC	CIDR3	RO	0x000000B1	32	css600_apbrom_gpr Component Identification Register 3, CIDR3

10.8.2 Register descriptions

This section describes the `css600_apbrom_gpr` registers.

css600_apbrom_gpr register summary provides cross references to individual registers.

Note

The ROM table has a configuration parameter `TIE_OFF_PRESENT`, which, if set to 1, allows for the selective removal of any entry using the entry_present input bus. The entry_present bus contains one bit per ROM table entry. If a given entry_present[n] bit is tied HIGH, then the value in ROMEntry<n>.PRESENT[1:0] is taken directly from the value in the `ROM_ENTRY<n>` parameter for that entry. If the entry_present[n] input is tied LOW, then a value of `0x3` in ROMEntry<n>.PRESENT[1:0] is modified to read `0x2` to indicate that the value is not present and is not the last entry.

10.8.2.1 css600_apbrom_gpr ROM Entries register 0, ROMEntry0

The ROMEntry register contains a descripter of a CoreSight component in the system. All ROM table entries conform to the same format.

Attributes

Offset

`0x0000`

Type

Read-only

Reset

`0x--------`

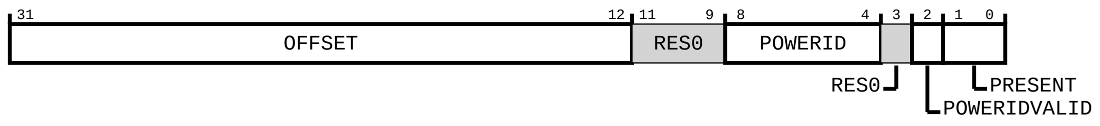
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-149: css600_apbrom_gpr_ROMEntry0



The following table shows the bit assignments.

Table 10-156: ROMEntry0 attributes

Bits	Reset value	Name	Type	Function
[31:12]	**IMPLEMENTATION DEFINED**	OFFSET	Read-only	The component address, relative to the base address of this ROM table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). If a component occupies more than a single 4KB block, OFFSET points to the 4KB block which contains the Peripheral ID and Component ID registers for the component. Negative values of OFFSET are permitted, using two's complement.
[11:9]	**IMPLEMENTATION DEFINED**	RES0	Read-only	Reserved bit or field with SBZP behavior
[8:4]	**IMPLEMENTATION DEFINED**	POWERID	Read-only	Indicates the power domain ID of the component. Only valid if bit 2 is set. If bit 2 is clear then this field has a value of 0. Possible values are 0 to 31, representing the 32 DBGPWRUPREQ/ACK interface pins of the component.
[3]	**IMPLEMENTATION DEFINED**	RES0	Read-only	Reserved bit or field with SBZP behavior
[2]	**IMPLEMENTATION DEFINED**	POWERIDVALID	Read-only	Indicates whether there is a power domain ID specified in the ROM table entry: **0** POWERID field of this register is not valid **1** POWERID field of this register is valid
[1:0]	**IMPLEMENTATION DEFINED**	PRESENT	Read-only	Indicates whether the ROM table entry is present: **0x0** ROM table entry not present. This is the last entry. **0x1** Reserved **0x2** ROM table entry not present. This is not the last entry. **0x3** ROM table entry present

10.8.2.2 css600_apbrom_gpr ROM Entries register 1, ROMEntry1

The ROMEntry register contains a descripter of a CoreSight component in the system. All ROM table entries conform to the same format.

Attributes

Offset

`0x0004`

Type

Read-only

Reset

`0x--------`

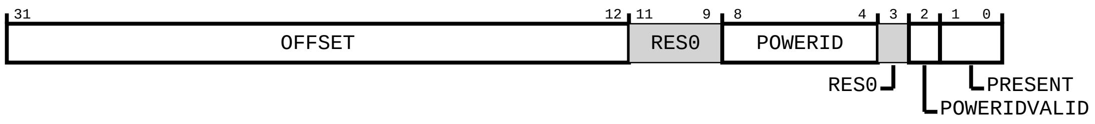
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-150: css600_apbrom_gpr_ROMEntry1



The following table shows the bit assignments.

Table 10-157: ROMEntry1 attributes

Bits	Reset value	Name	Type	Function
[31:12]	IMPLEMENTATION DEFINED	OFFSET	Read-only	The component address, relative to the base address of this ROM table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). If a component occupies more than a single 4KB block, OFFSET points to the 4KB block which contains the Peripheral ID and Component ID registers for the component. Negative values of OFFSET are permitted, using two's complement.
[11:9]	IMPLEMENTATION DEFINED	RES0	Read-only	Reserved bit or field with SBZP behavior
[8:4]	IMPLEMENTATION DEFINED	POWERID	Read-only	Indicates the power domain ID of the component. Only valid if bit 2 is set. If bit 2 is clear then this field has a value of 0. Possible values are 0 to 31, representing the 32 DBGPWRUPREQ/ACK interface pins of the component.
[3]	IMPLEMENTATION DEFINED	RES0	Read-only	Reserved bit or field with SBZP behavior
[2]	IMPLEMENTATION DEFINED	POWERIDVALID	Read-only	Indicates whether there is a power domain ID specified in the ROM table entry: **0** POWERID field of this register is not valid **1** POWERID field of this register is valid
[1:0]	IMPLEMENTATION DEFINED	PRESENT	Read-only	Indicates whether the ROM table entry is present: **0x0** ROM table entry not present. This is the last entry. **0x1** Reserved **0x2** ROM table entry not present. This is not the last entry. **0x3** ROM table entry present

10.8.2.3 css600_apbrom_gpr ROM Entries register 2, ROMEntry2

The ROMEntry register contains a descripter of a CoreSight component in the system. All ROM table entries conform to the same format.

Attributes

Offset

`0x0008`

Type

Read-only

Reset

`0x--------`

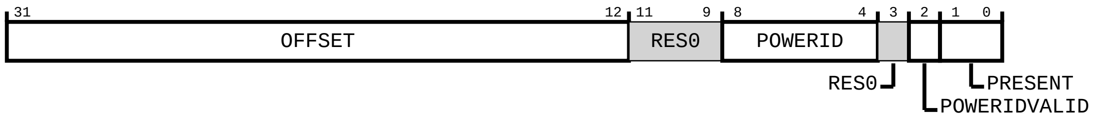
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-151: css600_apbrom_gpr_ROMEntry2



The following table shows the bit assignments.

Table 10-158: ROMEntry2 attributes

Bits	Reset value	Name	Type	Function
[31:12]	**IMPLEMENTATION DEFINED**	OFFSET	Read-only	The component address, relative to the base address of this ROM table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). If a component occupies more than a single 4KB block, OFFSET points to the 4KB block which contains the Peripheral ID and Component ID registers for the component. Negative values of OFFSET are permitted, using two's complement.
[11:9]	**IMPLEMENTATION DEFINED**	RES0	Read-only	Reserved bit or field with SBZP behavior
[8:4]	**IMPLEMENTATION DEFINED**	POWERID	Read-only	Indicates the power domain ID of the component. Only valid if bit 2 is set. If bit 2 is clear then this field has a value of 0. Possible values are 0 to 31, representing the 32 DBGPWRUPREQ/ACK interface pins of the component.
[3]	**IMPLEMENTATION DEFINED**	RES0	Read-only	Reserved bit or field with SBZP behavior
[2]	**IMPLEMENTATION DEFINED**	POWERIDVALID	Read-only	Indicates whether there is a power domain ID specified in the ROM table entry: **0** POWERID field of this register is not valid **1** POWERID field of this register is valid
[1:0]	**IMPLEMENTATION DEFINED**	PRESENT	Read-only	Indicates whether the ROM table entry is present: **`0x0`** ROM table entry not present. This is the last entry. **`0x1`** Reserved **`0x2`** ROM table entry not present. This is not the last entry. **`0x3`** ROM table entry present

10.8.2.4 css600_apbrom_gpr ROM Entries register 511, ROMEntry511

The ROMEntry register contains a descripter of a CoreSight component in the system. All ROM table entries conform to the same format.

Attributes

Offset

`0x07FC`

Type

Read-only

Reset

`0x--------`

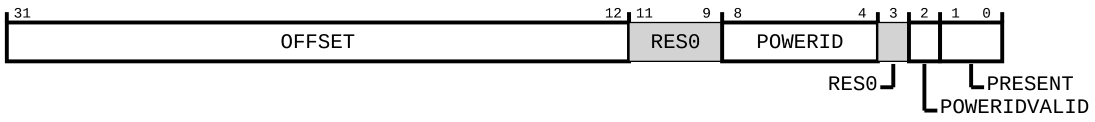
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-152: css600_apbrom_gpr_ROMEntry511



The following table shows the bit assignments.

Table 10-159: ROMEntry511 attributes

Bits	Reset value	Name	Type	Function
[31:12]	**IMPLEMENTATION DEFINED**	OFFSET	Read-only	The component address, relative to the base address of this ROM table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). If a component occupies more than a single 4KB block, OFFSET points to the 4KB block which contains the Peripheral ID and Component ID registers for the component. Negative values of OFFSET are permitted, using two's complement.
[11:9]	**IMPLEMENTATION DEFINED**	RES0	Read-only	Reserved bit or field with SBZP behavior
[8:4]	**IMPLEMENTATION DEFINED**	POWERID	Read-only	Indicates the power domain ID of the component. Only valid if bit 2 is set. If bit 2 is clear then this field has a value of 0. Possible values are 0 to 31, representing the 32 DBGPWRUPREQ/ACK interface pins of the component.
[3]	**IMPLEMENTATION DEFINED**	RES0	Read-only	Reserved bit or field with SBZP behavior
[2]	**IMPLEMENTATION DEFINED**	POWERIDVALID	Read-only	Indicates whether there is a power domain ID specified in the ROM table entry: **0** POWERID field of this register is not valid **1** POWERID field of this register is valid

Bits	Reset value	Name	Type	Function
[1:0]	IMPLEMENTATION DEFINED	PRESENT	Read-only	Indicates whether the ROM table entry is present: **0x0** ROM table entry not present. This is the last entry. **0x1** Reserved **0x2** ROM table entry not present. This is not the last entry. **0x3** ROM table entry present

10.8.2.5 css600_apbrom_gpr Debug Power Control Register 0, DBGPCR0

The DBGPCRn registers indicate whether power has been requested for a debug domain.

Attributes

Offset

`0x0A00`

Type

Read-write

Reset

`0x00000001`

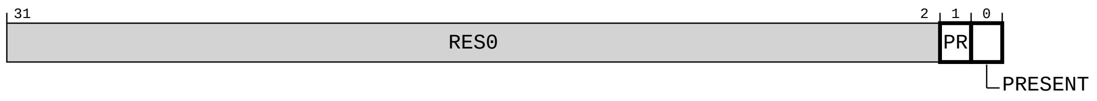
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-153: css600_apbrom_gpr_DBGPCR0



The following table shows the bit assignments.

Table 10-160: DBGPCR0 attributes

Bits	Reset value	Name	Type	Function
[31:2]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[1]	`0b0`	PR	Read-write	Power request. Reserved if DBGPCRn.PRESENT=0 or SYSPCRn=0. **0** Indicates that power is not requested for debug domain 0 **1** Indicates that power is requested for debug domain 0
[0]	`0b1`	PRESENT	Read-only	Indicates the presence of power domain control for debug domain 0: **0** Indicates that the power request is not implemented for debug domain 0 **1** Indicates that the power request is implemented for debug domain 0

10.8.2.6 css600_apbrom_gpr Debug Power Control Register 1, DBGPCR1

The DBGPCRn registers indicate whether power has been requested for a debug domain.

Attributes

Offset

`0x0A04`

Type

Read-write

Reset

`0x0000000-`

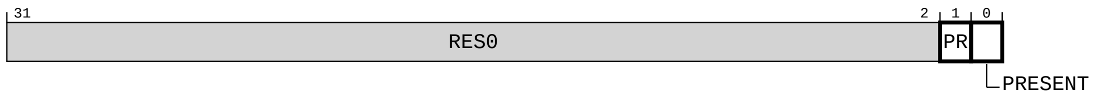
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-154: css600_apbrom_gpr_DBGPCR1



The following table shows the bit assignments.

Table 10-161: DBGPCR1 attributes

Bits	Reset value	Name	Type	Function
[31:2]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[1]	`0b0`	PR	Read-write	Power request. Reserved if DBGPCRn.PRESENT=0 or SYSPCRn=0. **0** Indicates that power is not requested for debug domain 1 **1** Indicates that power is requested for debug domain 1
[0]	IMPLEMENTATION DEFINED	PRESENT	Read-only	Indicates the presence of power domain control for debug domain 1: **0** Indicates that the power request is not implemented for debug domain 1 **1** Indicates that the power request is implemented for debug domain 1

10.8.2.7 css600_apbrom_gpr Debug Power Control Register 2, DBGPCR2

The DBGPCRn registers indicate whether power has been requested for a debug domain.

Attributes

Offset

`0x0A08`

Type

Read-write

Reset

`0x0000000-`

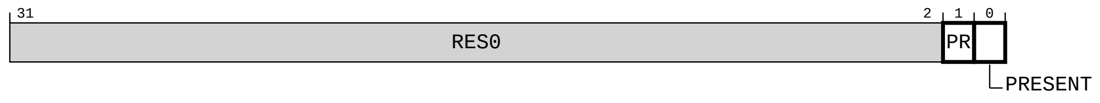
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-155: css600_apbrom_gpr_DBGPCR2



The following table shows the bit assignments.

Table 10-162: DBGPCR2 attributes

Bits	Reset value	Name	Type	Function
[31:2]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[1]	`0b0`	PR	Read-write	Power request. Reserved if DBGPCRn.PRESENT=0 or SYSPCRn=0. **0** Indicates that power is not requested for debug domain 2 **1** Indicates that power is requested for debug domain 2
[0]	IMPLEMENTATION DEFINED	PRESENT	Read-only	Indicates the presence of power domain control for debug domain 2: **0** Indicates that the power request is not implemented for debug domain 2 **1** Indicates that the power request is implemented for debug domain 2

10.8.2.8 css600_apbrom_gpr Debug Power Control Register 31, DBGPCR31

The DBGPCRn registers indicate whether power has been requested for a debug domain.

Attributes

Offset

`0x0A7C`

Type

Read-write

Reset

`0x0000000-`

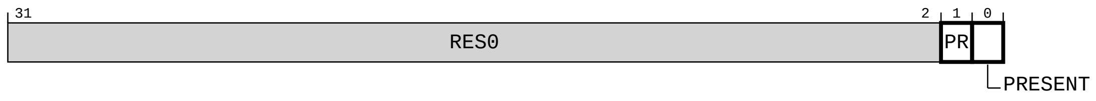
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-156: css600_apbrom_gpr_DBGPCR31



The following table shows the bit assignments.

Table 10-163: DBGPCR31 attributes

Bits	Reset value	Name	Type	Function
[31:2]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[1]	`0b0`	PR	Read-write	Power request. Reserved if DBGPCRn.PRESENT=0 or SYSPCRn=0. **0** Indicates that power is not requested for debug domain 31 **1** Indicates that power is requested for debug domain 31
[0]	IMPLEMENTATION DEFINED	PRESENT	Read-only	Indicates the presence of power domain control for debug domain 31: **0** Indicates that the power request is not implemented for debug domain 31 **1** Indicates that the power request is implemented for debug domain 31

10.8.2.9 css600_apbrom_gpr Debug Power Status Register 0, DBGPSR0

The DBGPSRn registers indicate the power status for a debug domain.

Attributes

Offset

`0x0A80`

Type

Read-only

Reset

`0x0000000-`

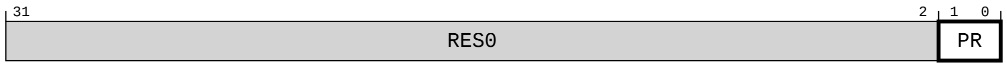
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-157: css600_apbrom_gpr_DBGPSR0



The following table shows the bit assignments.

Table 10-164: DBGPSR0 attributes

Bits	Reset value	Name	Type	Function
[31:2]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[1:0]	UNKNOWN	PR	Read-only	Power status. **`0x0`** Debug domain n might not be powered. **`0x3`** Debug domain n is powered and must remain powered until DBGPCRn.PR=0.

10.8.2.10 css600_apbrom_gpr Debug Power Status Register 1, DBGPSR1

The DBGPSRn registers indicate the power status for a debug domain.

Attributes

Offset

`0x0A84`

Type

Read-only

Reset

`0x0000000-`

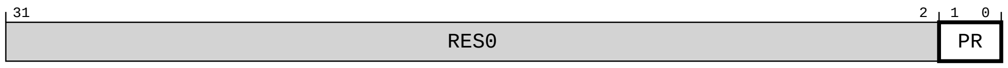
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-158: css600_apbrom_gpr_DBGPSR1



The following table shows the bit assignments.

Table 10-165: DBGPSR1 attributes

Bits	Reset value	Name	Type	Function
[31:2]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[1:0]	UNKNOWN	PR	Read-only	Power status. **`0x0`** Debug domain n might not be powered. **`0x3`** Debug domain n is powered and must remain powered until DBGPCRn.PR=0.

10.8.2.11 css600_apbrom_gpr Debug Power Status Register 2, DBGPSR2

The DBGPSRn registers indicate the power status for a debug domain.

Attributes

Offset

`0x0A88`

Type

Read-only

Reset

`0x0000000-`

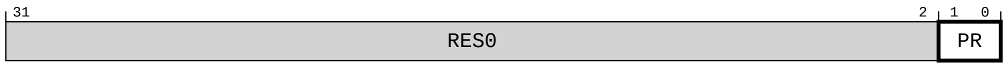
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-159: css600_apbrom_gpr_DBGPSR2



The following table shows the bit assignments.

Table 10-166: DBGPSR2 attributes

Bits	Reset value	Name	Type	Function
[31:2]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[1:0]	UNKNOWN	PR	Read-only	Power status. **`0x0`** Debug domain n might not be powered. **`0x3`** Debug domain n is powered and must remain powered until DBGPCRn.PR=0.

10.8.2.12 css600_apbrom_gpr Debug Power Status Register 31, DBGPSR31

The DBGPSRn registers indicate the power status for a debug domain.

Attributes

Offset

`0x0AFC`

Type

Read-only

Reset

`0x0000000-`

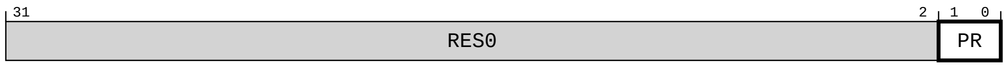
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-160: css600_apbrom_gpr_DBGPSR31



The following table shows the bit assignments.

Table 10-167: DBGPSR31 attributes

Bits	Reset value	Name	Type	Function
[31:2]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior

Bits	Reset value	Name	Type	Function
[1:0]	UNKNOWN	PR	Read-only	Power status. **0x0** Debug domain n might not be powered. **0x3** Debug domain n is powered and must remain powered until DBGPCRn.PR=0.

10.8.2.13 css600_apbrom_gpr System Power Control Register 0, SYSPCR0

The SYSPCRn registers indicate whether power has been requested for a system domain.

Attributes

Offset

`0x0B00`

Type

Read-write

Reset

`0x00000001`

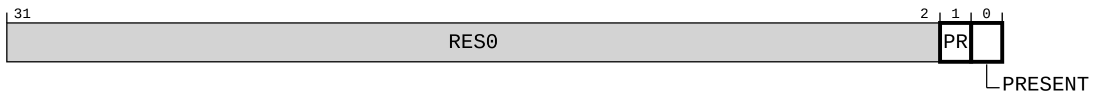
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-161: css600_apbrom_gpr_SYSPCR0



The following table shows the bit assignments.

Table 10-168: SYSPCR0 attributes

Bits	Reset value	Name	Type	Function
[31:2]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[1]	`0b0`	PR	Read-write	Power request. Reserved if DBGPCRn.PRESENT=0 or SYSPCRn=0. **0** Indicates that power is not requested for system domain 0 **1** Indicates that power is requested for system domain 0
[0]	`0b1`	PRESENT	Read-only	Indicates the presence of power domain control for system domain 0 **0** Indicates that the power request is not implemented for system domain 0 **1** Indicates that the power request is implemented for system domain 0

10.8.2.14 css600_apbrom_gpr System Power Control Register 1, SYSPCR1

The SYSPCRn registers indicate whether power has been requested for a system domain.

Attributes

Offset

`0x0B04`

Type

Read-write

Reset

`0x0000000-`

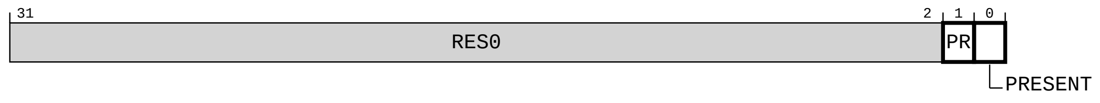
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-162: css600_apbrom_gpr_SYSPCR1



The following table shows the bit assignments.

Table 10-169: SYSPCR1 attributes

Bits	Reset value	Name	Type	Function
[31:2]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[1]	`0b0`	PR	Read-write	Power request. Reserved if DBGPCRn.PRESENT=0 or SYSPCRn=0. **0** Indicates that power is not requested for system domain 1 **1** Indicates that power is requested for system domain 1
[0]	IMPLEMENTATION DEFINED	PRESENT	Read-only	Indicates the presence of power domain control for system domain 1 **0** Indicates that the power request is not implemented for system domain 1 **1** Indicates that the power request is implemented for system domain 1

10.8.2.15 css600_apbrom_gpr System Power Control Register 2, SYSPCR2

The SYSPCRn registers indicate whether power has been requested for a system domain.

Attributes

Offset

`0x0B08`

Type

Read-write

Reset

`0x0000000-`

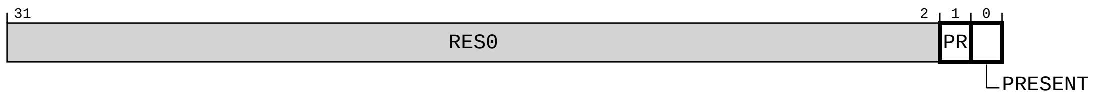
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-163: css600_apbrom_gpr_SYSPCR2



The following table shows the bit assignments.

Table 10-170: SYSPCR2 attributes

Bits	Reset value	Name	Type	Function
[31:2]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[1]	`0b0`	PR	Read-write	Power request. Reserved if DBGPCRn.PRESENT=0 or SYSPCRn=0. **0** Indicates that power is not requested for system domain 2 **1** Indicates that power is requested for system domain 2
[0]	IMPLEMENTATION DEFINED	PRESENT	Read-only	Indicates the presence of power domain control for system domain 2 **0** Indicates that the power request is not implemented for system domain 2 **1** Indicates that the power request is implemented for system domain 2

10.8.2.16 css600_apbrom_gpr System Power Control Register 31, SYSPCR31

The SYSPCRn registers indicate whether power has been requested for a system domain.

Attributes

Offset

`0x0B7C`

Type

Read-write

Reset

`0x0000000-`

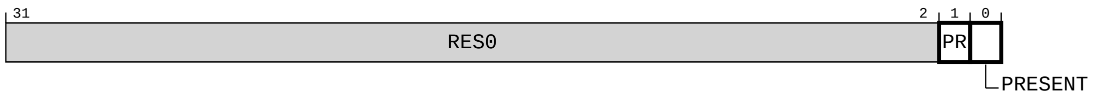
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-164: css600_apbrom_gpr_SYSPCR31



The following table shows the bit assignments.

Table 10-171: SYSPCR31 attributes

Bits	Reset value	Name	Type	Function
[31:2]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[1]	`0b0`	PR	Read-write	Power request. Reserved if DBGPCRn.PRESENT=0 or SYSPCRn=0. **0** Indicates that power is not requested for system domain 31 **1** Indicates that power is requested for system domain 31
[0]	IMPLEMENTATION DEFINED	PRESENT	Read-only	Indicates the presence of power domain control for system domain 31 **0** Indicates that the power request is not implemented for system domain 31 **1** Indicates that the power request is implemented for system domain 31

10.8.2.17 css600_apbrom_gpr System Power Status Register 0, SYSPSR0

The SYSPSRn registers indicate the power status for a system domain.

Attributes

Offset

`0x0B80`

Type

Read-only

Reset

`0x0000000-`

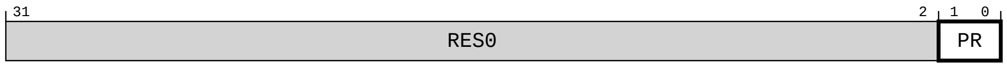
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-165: css600_apbrom_gpr_SYSPSR0



The following table shows the bit assignments.

Table 10-172: SYSPSR0 attributes

Bits	Reset value	Name	Type	Function
[31:2]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[1:0]	UNKNOWN	PR	Read-only	Power status: **`0x0`** System domain n might not be powered **`0x3`** System domain n is powered and must remain powered until DBGPCRn.PR=0

10.8.2.18 css600_apbrom_gpr System Power Status Register 1, SYSPSR1

The SYSPSRn registers indicate the power status for a system domain.

Attributes

Offset

`0x0B84`

Type

Read-only

Reset

`0x0000000-`

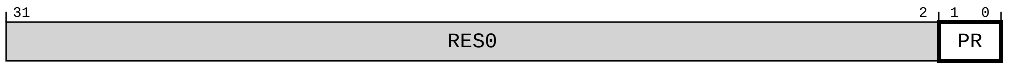
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-166: css600_apbrom_gpr_SYSPSR1



The following table shows the bit assignments.

Table 10-173: SYSPSR1 attributes

Bits	Reset value	Name	Type	Function
[31:2]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[1:0]	UNKNOWN	PR	Read-only	Power status: **`0x0`** System domain n might not be powered **`0x3`** System domain n is powered and must remain powered until DBGPCRn.PR=0

10.8.2.19 css600_apbrom_gpr System Power Status Register 2, SYSPSR2

The SYSPSRn registers indicate the power status for a system domain.

Attributes

Offset

`0x0B88`

Type

Read-only

Reset

`0x0000000-`

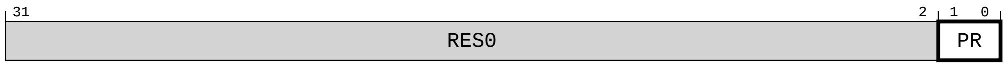
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-167: css600_apbrom_gpr_SYSPSR2



The following table shows the bit assignments.

Table 10-174: SYSPSR2 attributes

Bits	Reset value	Name	Type	Function
[31:2]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[1:0]	UNKNOWN	PR	Read-only	Power status: **`0x0`** System domain n might not be powered **`0x3`** System domain n is powered and must remain powered until DBGPCRn.PR=0

10.8.2.20 css600_apbrom_gpr System Power Status Register 31, SYSPSR31

The SYSPSRn registers indicate the power status for a system domain.

Attributes

Offset

`0x0BFC`

Type

Read-only

Reset

`0x0000000-`

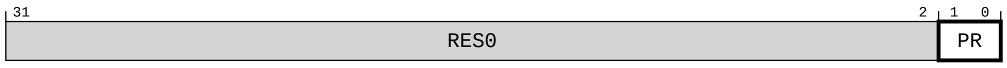
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-168: css600_apbrom_gpr_SYSPSR31



The following table shows the bit assignments.

Table 10-175: SYSPSR31 attributes

Bits	Reset value	Name	Type	Function
[31:2]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior

Bits	Reset value	Name	Type	Function
[1:0]	UNKNOWN	PR	Read-only	Power status: **0x0** System domain n might not be powered **0x3** System domain n is powered and must remain powered until DBGPCRn.PR=0

10.8.2.21 css600_apbrom_gpr Power Request ID Register, PRIDR0

The PRIDR0 register indicates the version of the power request function.

Attributes

Offset

`0x0C00`

Type

Read-only

Reset

`0x00000031`

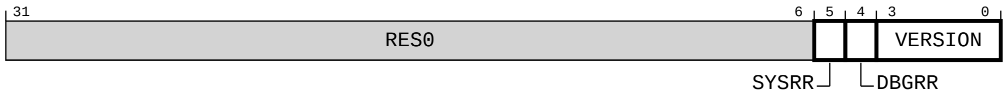
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-169: css600_apbrom_gpr_PRIDR0



The following table shows the bit assignments.

Table 10-176: PRIDR0 attributes

Bits	Reset value	Name	Type	Function
[31:6]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[5]	`0b1`	SYSRR	Read-only	Indicates whether the system reset request functionality is present **0** System reset request functionality is not implemented. **1** System reset request functionality is implemented. SYSRSTRR and SYSRSTAR are both implemented.

Bits	Reset value	Name	Type	Function
[4]	`0b1`	DBGRR	Read-only	Indicates whether the debug reset request functionality is present: **0** Debug reset request functionality is not implemented. **1** Debug reset request functionality is implemented. DBGRSTRR and DBGRSTAR are both implemented.
[3:0]	`0b0001`	VERSION	Read-only	Version of the power request function. Set according to the GPR_PRESENT parameter. **`0x0`** Power request functionality is not included **`0x1`** Power request functionality version 1 is included

10.8.2.22 css600_apbrom_gpr Debug Reset Request Register, DBGRSTRR

The DBGRSTRR register indicates the status of a debug reset request.

Attributes

Offset

`0x0C10`

Type

Read-write

Reset

`0x00000000`

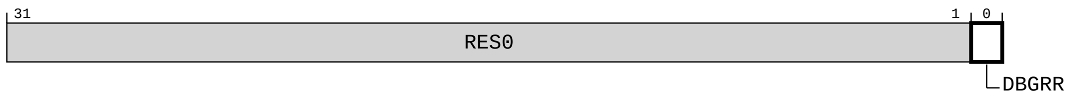
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-170: css600_apbrom_gpr_DBGRSTRR



The following table shows the bit assignments.

Table 10-177: DBGRSTRR attributes

Bits	Reset value	Name	Type	Function
[31:1]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior

Bits	Reset value	Name	Type	Function
[0]	`0b0`	DBGRR	Read-write	Debug reset request. The software needs to clear this bit after setting it. There is no automatic mechanism to clear it. **0** No reset is requested. cdbgrstreq output is LOW. **1** Reset is requested. cdbgrstreq output is HIGH.

10.8.2.23 css600_apbrom_gpr Debug Reset Acknowledge Register, DBGRSTAR

The DBGRSTAR register acknowledges a debug reset request.

Attributes

Offset

`0x0C14`

Type

Read-only

Reset

`0x00000000`

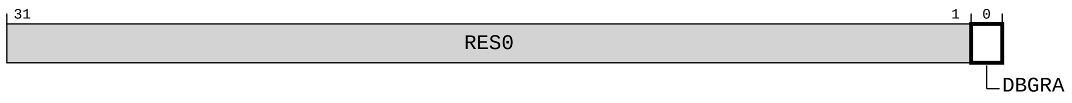
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-171: css600_apbrom_gpr_DBGRSTAR



The following table shows the bit assignments.

Table 10-178: DBGRSTAR attributes

Bits	Reset value	Name	Type	Function
[31:1]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[0]	`0b0`	DBGRA	Read-only	Debug reset acknowledge: **0** No reset is requested or reset is not acknowledged **1** Reset is acknowledged by the external reset controller

10.8.2.24 css600_apbrom_gpr System Reset Request Register, SYSRSTRR

The SYSRSTRR register indicates the status of a system reset request.

Attributes

Offset

`0x0C18`

Type

Read-write

Reset

`0x00000000`

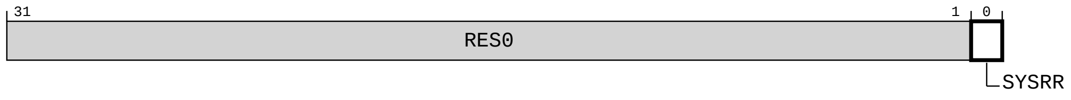
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-172: css600_apbrom_gpr_SYSRSTRR



The following table shows the bit assignments.

Table 10-179: SYSRSTRR attributes

Bits	Reset value	Name	Type	Function
[31:1]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[0]	`0b0`	SYSRR	Read-write	System reset request. The software needs to clear this bit after setting it. There is no automatic mechanism to clear it. **0** No reset is requested. csysrstreq output is LOW. **1** Reset is requested. csysrstreq output is HIGH.

10.8.2.25 css600_apbrom_gpr System Reset Acknowledge Register, SYSRSTAR

The SYSRSTAR register acknowledges a system reset request.

Attributes

Offset

`0x0C1C`

Type

Read-only

Reset

`0x00000000`

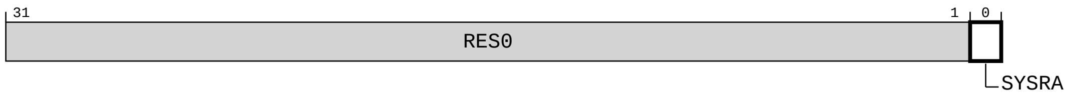
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-173: css600_apbrom_gpr_SYSRSTAR



The following table shows the bit assignments.

Table 10-180: SYSRSTAR attributes

Bits	Reset value	Name	Type	Function
[31:1]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[0]	`0b0`	SYSRA	Read-only	System reset acknowledge: **0** No reset is requested or reset is not acknowledged **1** Reset is acknowledged by the external reset controller

10.8.2.26 css600_apbrom_gpr Authentication Status Register, AUTHSTATUS

Reports the current status of the authentication control signals.

Attributes

Offset

`0x0FB8`

Type

Read-only

Reset

`0x000000--`

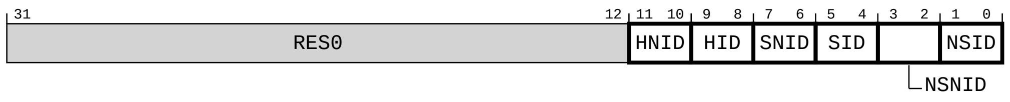
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-174: css600_apbrom_gpr_AUTHSTATUS



The following table shows the bit assignments.

Table 10-181: AUTHSTATUS attributes

Bits	Reset value	Name	Type	Function
[31:12]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[11:10]	`0b00`	HNID	Read-only	Hypervisor non-invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.
[9:8]	`0b00`	HID	Read-only	Hypervisor invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.
[7:6]	UNKNOWN	SNID	Read-only	Secure non-invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.
[5:4]	UNKNOWN	SID	Read-only	Secure invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.
[3:2]	UNKNOWN	NSNID	Read-only	Non-secure non-invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.
[1:0]	UNKNOWN	NSID	Read-only	Non-secure invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.

10.8.2.27 css600_apbrom_gpr Device Architecture Register, DEVARCH

Identifies the architect and architecture of a CoreSight component. The architect might differ from the designer of a component, for example Arm defines the architecture but another company designs and implements the component.

Attributes

Offset

`0x0FBC`

Type

Read-only

Reset

`0x47700AF7`

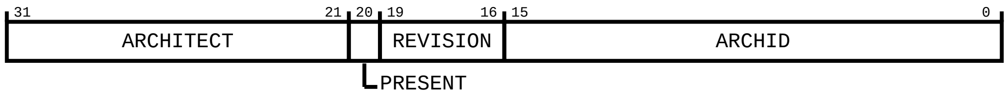
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-175: css600_apbrom_gpr_DEVARCH



The following table shows the bit assignments.

Table 10-182: DEVARCH attributes

Bits	Reset value	Name	Type	Function
[31:21]	`0x23B`	ARCHITECT	Read-only	Returns 0x23B, denoting Arm as architect of the component
[20]	`0b1`	PRESENT	Read-only	Returns 1, indicating that the DEVARCH register is present
[19:16]	`0b0000`	REVISION	Read-only	Architecture revision. Returns the revision of the architecture that the ARCHID field specifies.
[15:0]	`0xAF7`	ARCHID	Read-only	Architecture ID. Returns 0x0AF7, identifying ROM Table Architecture v0.

10.8.2.28 css600_apbrom_gpr Device Configuration Register, DEVID

This register is **IMPLEMENTATION DEFINED** for each Part Number and Designer. The register indicates the capabilities of the component.

Attributes

Offset

`0x0FC8`

Type

Read-only

Reset

`0x000000-0`

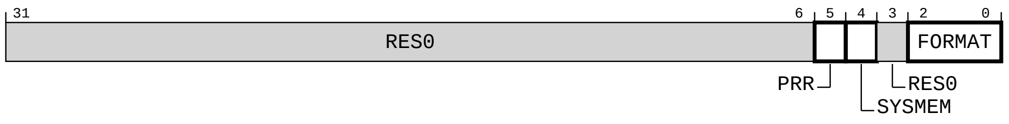
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-176: css600_apbrom_gpr_DEVID



The following table shows the bit assignments.

Table 10-183: DEVID attributes

Bits	Reset value	Name	Type	Function
[31:6]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[5]	`0b0`	PRR	Read-only	Indicates that power request functionality is included. Set by the GPR_PRESENT parameter. **0** GPR is not included (css600_apbrom) **1** GPR is included (css600_apbrom_gpr)
[4]	IMPLEMENTATION DEFINED	SYSMEM	Read-only	Indicates whether system memory is present on the bus. Set by the SYSMEM parameter. **0** System memory is not present and the bus is a dedicated debug bus **1** Indicates that there is system memory on the bus
[3]	`0b0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[2:0]	`0b000`	FORMAT	Read-only	Indicates that this is a 32-bit ROM table.

10.8.2.29 css600_apbrom_gpr Peripheral Identification Register 4, PIDR4

The PIDR4 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FD0`

Type

Read-only

Reset

`0x0000000-`

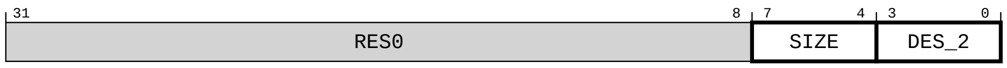
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-177: css600_apbrom_gpr_PIDR4



The following table shows the bit assignments.

Table 10-184: PIDR4 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0000`	SIZE	Read-only	Indicates the memory size that is used by this component. Returns 0 indicating that the component uses an **UNKNOWN** number of 4KB blocks. Using the SIZE field to indicate the size of the component is deprecated.
[3:0]	**IMPLEMENTATION DEFINED**	DES_2	Read-only	JEP106 continuation code. Together, with PIDR2.DES_1 and PIDR1.DES_0, they indicate the designer of the component and not the implementer, except where the two are the same.

10.8.2.30 css600_apbrom_gpr Peripheral Identification Register 5, PIDR5

The PIDR5 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FD4`

Type

Read-only

Reset

`0x00000000`

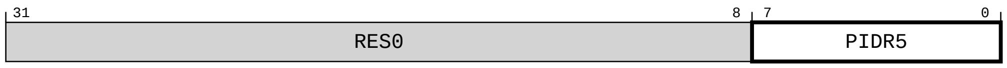
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-178: css600_apbrom_gpr_PIDR5



The following table shows the bit assignments.

Table 10-185: PIDR5 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	PIDR5	Read-only	Reserved.

10.8.2.31 css600_apbrom_gpr Peripheral Identification Register 6, PIDR6

The PIDR6 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FD8`

Type

Read-only

Reset

`0x00000000`

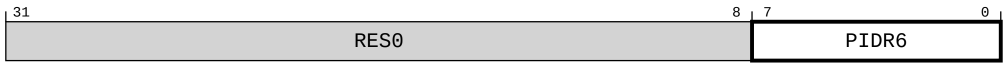
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-179: css600_apbrom_gpr_PIDR6



The following table shows the bit assignments.

Table 10-186: PIDR6 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	PIDR6	Read-only	Reserved.

10.8.2.32 css600_apbrom_gpr Peripheral Identification Register 7, PIDR7

The PIDR7 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FDC`

Type

Read-only

Reset

`0x00000000`

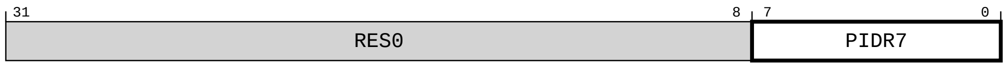
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-180: css600_apbrom_gpr_PIDR7



The following table shows the bit assignments.

Table 10-187: PIDR7 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	PIDR7	Read-only	Reserved.

10.8.2.33 css600_apbrom_gpr Peripheral Identification Register 0, PIDR0

The PIDR0 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FE0`

Type

Read-only

Reset

`0x000000--`

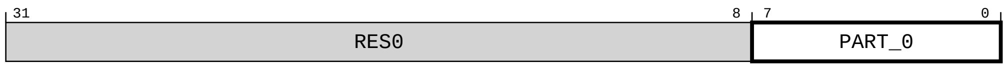
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-181: css600_apbrom_gpr_PIDR0



The following table shows the bit assignments.

Table 10-188: PIDR0 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	IMPLEMENTATION DEFINED	PART_0	Read-only	Part number, bits[7:0]. Set by the configuration inputs part_number[7:0]

10.8.2.34 css600_apbrom_gpr Peripheral Identification Register 1, PIDR1

The PIDR1 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FE4`

Type

Read-only

Reset

`0x000000FF`

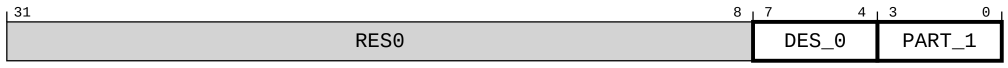
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-182: css600_apbrom_gpr_PIDR1



The following table shows the bit assignments.

Table 10-189: PIDR1 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	IMPLEMENTATION DEFINED	DES_0	Read-only	JEP106 identification code, bits[3:0]. Set by the configuration inputs jep106_id[3:0]. Together, with PIDR4.DES_2 and PIDR2.DES_1, they indicate the designer of the component and not the implementer, except where the two are the same.
[3:0]	IMPLEMENTATION DEFINED	PART_1	Read-only	Part number, bits[11:8]. Set by the configuration inputs part_number[11:8].

10.8.2.35 css600_apbrom_gpr Peripheral Identification Register 2, PIDR2

The PIDR2 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FE8`

Type

Read-only

Reset

`0x000000--`

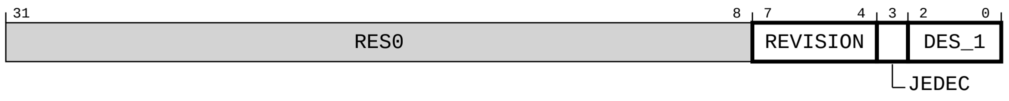
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-183: css600_apbrom_gpr_PIDR2



The following table shows the bit assignments.

Table 10-190: PIDR2 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	IMPLEMENTATION DEFINED	REVISION	Read-only	Revision. Set by the configuration inputs revision[3:0].
[3]	`0b1`	JEDEC	Read-only	1 - Always set. Indicates that a JEDEC assigned value is used.
[2:0]	IMPLEMENTATION DEFINED	DES_1	Read-only	JEP106 identification code, bits[6:4]. Set by the configuration inputs jep106_id[6:4]. Together, with PIDR4.DES_2 and PIDR1.DES_0, they indicate the designer of the component and not the implementer, except where the two are the same.

10.8.2.36 css600_apbrom_gpr Peripheral Identification Register 3, PIDR3

The PIDR3 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FEC`

Type

Read-only

Reset

`0x000000-0`

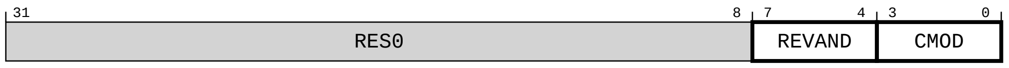
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-184: css600_apbrom_gpr_PIDR3



The following table shows the bit assignments.

Table 10-191: PIDR3 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0000`	REVAND	Read-only	This field indicates minor errata fixes specific to this design, for example metal fixes after implementation. In most cases this field is `0x0`.
[3:0]	`0b0000`	CMOD	Read-only	Customer Modified. Where the component is reusable IP, this value indicates if the customer has modified the behavior of the component. In most cases this field is `0x0`.

10.8.2.37 css600_apbrom_gpr Component Identification Register 0, CIDR0

The CIDR0 register is part of the set of component identification registers.

Attributes

Offset

`0x0FF0`

Type

Read-only

Reset

`0x0000000D`

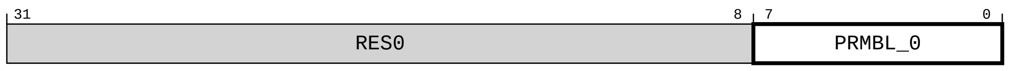
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-185: css600_apbrom_gpr_CIDR0



The following table shows the bit assignments.

Table 10-192: CIDR0 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	0xD	PRMBL_0	Read-only	Preamble. Returns `0x0D`.

10.8.2.38 css600_apbrom_gpr Component Identification Register 1, CIDR1

The CIDR1 register is part of the set of component identification registers.

Attributes

Offset

`0x0FF4`

Type

Read-only

Reset

`0x00000090`

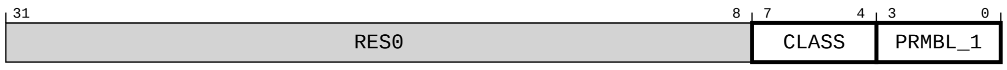
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-186: css600_apbrom_gpr_CIDR1



The following table shows the bit assignments.

Table 10-193: CIDR1 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b1001`	CLASS	Read-only	Component class. Returns `0x9`, indicating this is a CoreSight component.
[3:0]	`0b0000`	PRMBL_1	Read-only	Preamble. Returns `0x0`.

10.8.2.39 css600_apbrom_gpr Component Identification Register 2, CIDR2

The CIDR2 register is part of the set of component identification registers.

Attributes

Offset

`0x0FF8`

Type

Read-only

Reset

`0x00000005`

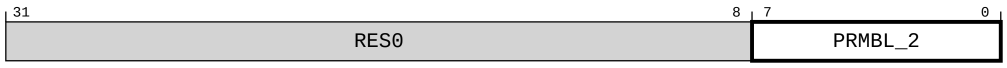
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-187: css600_apbrom_gpr_CIDR2



The following table shows the bit assignments.

Table 10-194: CIDR2 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x5`	PRMBL_2	Read-only	Preamble. Returns `0x05`.

10.8.2.40 css600_apbrom_gpr Component Identification Register 3, CIDR3

The CIDR3 register is part of the set of component identification registers.

Attributes

Offset

`0x0FFC`

Type

Read-only

Reset

`0x000000B1`

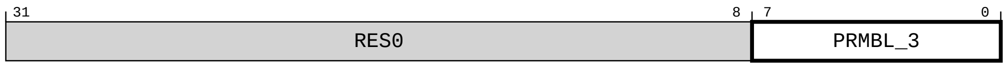
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-188: css600_apbrom_gpr_CIDR3



The following table shows the bit assignments.

Table 10-195: CIDR3 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0xB1`	PRMBL_3	Read-only	Preamble. Returns `0xB1`.

10.9 `css600_atbfunnel_prog` introduction

This section describes the programmers model of the `css600_atbfunnel_prog`.

10.9.1 css600_atbfunnel register summary

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

Note

A reset value containing one or more '-' means that this register contains **UNKNOWN** or **IMPLEMENTATION-DEFINED** values. See the relevant register description for more information.

Locations that are not listed in the table are Reserved.

Table 10-196: css600_atbfunnel register summary

Offset	Name	Type	Reset	Width	Description
`0x0000`	FUNNELCONTROL	RW	`0x00000300`	32	css600_atbfunnel Funnel Control register, FUNNELCONTROL
`0x0004`	PRIORITYCONTROL	RW	`0x00000000`	32	css600_atbfunnel Priority Control register, PRIORITYCONTROL
`0x0EEC`	ITATBDATA0	RW	`0x00000000`	32	css600_atbfunnel Integration test data register, ITATBDATA0
`0x0EF0`	ITATBCTR3	RW	`0x00000000`	32	css600_atbfunnel Integration test control register 3, ITATBCTR3
`0x0EF4`	ITATBCTR2	RW	`0x00000000`	32	css600_atbfunnel Integration test control register 2, ITATBCTR2
`0x0EF8`	ITATBCTR1	RW	`0x00000000`	32	css600_atbfunnel Integration test control register 1, ITATBCTR1
`0x0EFC`	ITATBCTR0	RW	`0x00000000`	32	css600_atbfunnel Integration test control register 0, ITATBCTR0
`0x0F00`	ITCTRL	RW	`0x00000000`	32	css600_atbfunnel Integration Mode Control Register, ITCTRL
`0x0FA0`	CLAIMSET	RW	`0x0000000F`	32	css600_atbfunnel Claim Tag Set Register, CLAIMSET
`0x0FA4`	CLAIMCLR	RW	`0x00000000`	32	css600_atbfunnel Claim Tag Clear Register, CLAIMCLR
`0x0FA8`	DEVAFF0	RO	`0x00000000`	32	css600_atbfunnel Device Affinity register 0, DEVAFF0
`0x0FAC`	DEVAFF1	RO	`0x00000000`	32	css600_atbfunnel Device Affinity register 1, DEVAFF1
`0x0FB8`	AUTHSTATUS	RO	`0x00000000`	32	css600_atbfunnel Authentication Status Register, AUTHSTATUS
`0x0FBC`	DEVARCH	RO	`0x00000000`	32	css600_atbfunnel Device Architecture Register, DEVARCH
`0x0FC0`	DEVID2	RO	`0x00000000`	32	css600_atbfunnel Device Configuration Register 2, DEVID2
`0x0FC4`	DEVID1	RO	`0x00000000`	32	css600_atbfunnel Device Configuration Register 1, DEVID1
`0x0FC8`	DEVID	RO	`0x0000003-`	32	css600_atbfunnel Device Configuration Register, DEVID
`0x0FCC`	DEVTYPE	RO	`0x00000012`	32	css600_atbfunnel Device Type Identifier Register, DEVTYPE
`0x0FD0`	PIDR4	RO	`0x00000004`	32	css600_atbfunnel Peripheral Identification Register 4, PIDR4

Offset	Name	Type	Reset	Width	Description
`0x0FD4`	PIDR5	RO	`0x00000000`	32	css600_atbfunnel Peripheral Identification Register 5, PIDR5
`0x0FD8`	PIDR6	RO	`0x00000000`	32	css600_atbfunnel Peripheral Identification Register 6, PIDR6
`0x0FDC`	PIDR7	RO	`0x00000000`	32	css600_atbfunnel Peripheral Identification Register 7, PIDR7
`0x0FE0`	PIDR0	RO	`0x000000EB`	32	css600_atbfunnel Peripheral Identification Register 0, PIDR0
`0x0FE4`	PIDR1	RO	`0x000000B9`	32	css600_atbfunnel Peripheral Identification Register 1, PIDR1
`0x0FE8`	PIDR2	RO	`0x0000003B`	32	css600_atbfunnel Peripheral Identification Register 2, PIDR2
`0x0FEC`	PIDR3	RO	`0x00000000`	32	css600_atbfunnel Peripheral Identification Register 3, PIDR3
`0x0FF0`	CIDR0	RO	`0x0000000D`	32	css600_atbfunnel Component Identification Register 0, CIDR0
`0x0FF4`	CIDR1	RO	`0x00000090`	32	css600_atbfunnel Component Identification Register 1, CIDR1
`0x0FF8`	CIDR2	RO	`0x00000005`	32	css600_atbfunnel Component Identification Register 2, CIDR2
`0x0FFC`	CIDR3	RO	`0x000000B1`	32	css600_atbfunnel Component Identification Register 3, CIDR3

10.9.2 Register descriptions

This section describes the `css600_atbfunnel_prog` registers.

css600_atbfunnel register summary provides cross references to individual registers.

10.9.2.1 css600_atbfunnel Funnel Control register, FUNNELCONTROL

The FUNNELCONTROL register enables each of the trace sources and controls the hold time for switching between them.

Attributes

Offset

`0x0000`

Type

Read-write

Reset

`0x00000300`

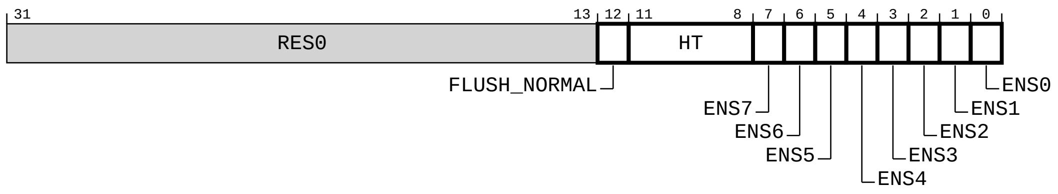
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-189: css600_atbfunnel_FUNNELCONTROL



The following table shows the bit assignments.

Table 10-197: FUNNELCONTROL attributes

Bits	Reset value	Name	Type	Function
[31:13]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[12]	`0b0`	FLUSH_NORMAL	Read-write	This bit, when clear, allows slave ports that are already flushed to receive further data even if there are other ports that have not completed flush. If set, a port that has completed flush is not be allowed to receive further data until all ports have completed flush.
[11:8]	`0b0011`	HT	Read-write	Hold time. Value sets the minimum hold time before switching trace sources (funnel inputs) based on the ID. Value used is programmed value + 1. **0x0** 1 transaction hold time **0x1** 2 transactions hold time **0x2** 3 transactions hold time **0x3** 4 transactions hold time **0x4** 5 transactions hold time **0x5** 6 transactions hold time **0x6** 7 transactions hold time **0x7** 8 transactions hold time **0x8** 9 transactions hold time **0x9** 10 transactions hold time 0xA 11 transactions hold time 0xB 12 transactions hold time 0xC 13 transactions hold time **0xD** 14 transactions hold time 0xE 15 transactions hold time 0xF Reserved
[7]	`0b0`	ENS7	Read-write	Enable slave interface 7. Field is RES0 if slave interface 7 is not implemented. 0 Slave interface disabled 1 Slave interface enabled
[6]	`0b0`	ENS6	Read-write	Enable slave interface 6. Field is RES0 if slave interface 6 is not implemented. 0 Slave interface disabled 1 Slave interface enabled
[5]	`0b0`	ENS5	Read-write	Enable slave interface 5. Field is RES0 if slave interface 5 is not implemented. 0 Slave interface disabled 1 Slave interface enabled

Bits	Reset value	Name	Type	Function
[4]	`0b0`	ENS4	Read-write	Enable slave interface 4. Field is RES0 if slave interface 4 is not implemented. **0** Slave interface disabled **1** Slave interface enabled
[3]	`0b0`	ENS3	Read-write	Enable slave interface 3. Field is RES0 if slave interface 3 is not implemented. **0** Slave interface disabled **1** Slave interface enabled
[2]	`0b0`	ENS2	Read-write	Enable slave interface 2. Field is RES0 if slave interface 2 is not implemented. **0** Slave interface disabled **1** Slave interface enabled
[1]	`0b0`	ENS1	Read-write	Enable slave interface 1. Field is RES0 if slave interface 1 is not implemented. **0** Slave interface disabled **1** Slave interface enabled
[0]	`0b0`	ENS0	Read-write	Enable slave interface 0. Field is RES0 if slave interface 0 is not implemented. **0** Slave interface disabled **1** Slave interface enabled

10.9.2.2 css600_atbfunnel Priority Control register, PRIORITYCONTROL

The PRIORITYCONTROL register sets the priority of each port (slave interface) of the funnel. The programming software requires that the ports are all disabled before the priority control register contents are changed. Changing the port priorities in real time is not supported. If the priority control register is written when one or more of the ports are enabled, then the write is silently rejected and the value in the priority control register remains unchanged. The lower the priority value, the higher is its priority when selecting the next port to be serviced. If two or more ports have the same priority value, then the lowest numbered port is serviced first.

Attributes

Offset

`0x0004`

Type

Read-write

Reset

`0x00000000`

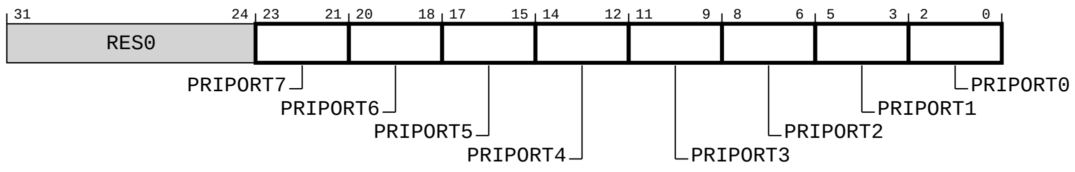
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-190: css600_atbfunnel_PRIORITYCONTROL



The following table shows the bit assignments.

Table 10-198: PRIORITYCONTROL attributes

Bits	Reset value	Name	Type	Function
[31:24]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[23:21]	`0b000`	PRIPORT7	Read-write	Priority value for port 7. Field is RES0 if port 7 is not implemented.
[20:18]	`0b000`	PRIPORT6	Read-write	Priority value for port 6. Field is RES0 if port 6 is not implemented.
[17:15]	`0b000`	PRIPORT5	Read-write	Priority value for port 5. Field is RES0 if port 5 is not implemented.
[14:12]	`0b000`	PRIPORT4	Read-write	Priority value for port 4. Field is RES0 if port 4 is not implemented.
[11:9]	`0b000`	PRIPORT3	Read-write	Priority value for port 3. Field is RES0 if port 3 is not implemented.
[8:6]	`0b000`	PRIPORT2	Read-write	Priority value for port 2. Field is RES0 if port 2 is not implemented.
[5:3]	`0b000`	PRIPORT1	Read-write	Priority value for port 1. Field is RES0 if port 1 is not implemented.
[2:0]	`0b000`	PRIPORT0	Read-write	Priority value for port 0. Field is RES0 if port 0 is not implemented.

10.9.2.3 css600_atbfunnel Integration test data register, ITATBDATA0

The ITATBDATA0 register allows observability and controllability of the ATDATA buses into and out of the funnel. For slave signals coming into the funnel, the register views the ports that are selected through the funnel control register. Only one port must be selected for integration test.

Attributes

Offset

`0x0EEC`

Type

Read-write

Reset

`0x00000000`

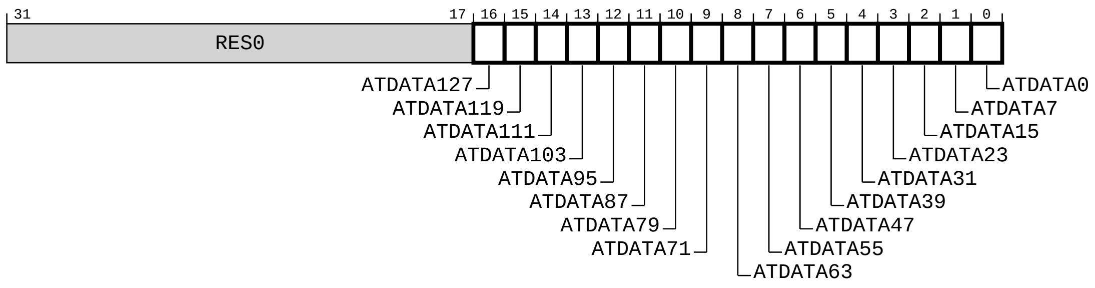
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-191: css600_atbfunnel_ITATBDATA0



The following table shows the bit assignments.

Table 10-199: ITATBDATA0 attributes

Bits	Reset value	Name	Type	Function
[31:17]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[16]	`0b0`	ATDATA127	Read-write	Reads atdata_s[127] and writes atdata_m[127]: **0** On reads, the value of atdata_s[127] is 0. On writes, sets atdata_m[127] to 0 **1** On reads, the value of atdata_s[127] is 1. On writes, sets atdata_m[127] to 1
[15]	`0b0`	ATDATA119	Read-write	Reads atdata_s[119] and writes atdata_m[119]: **0** On reads, the value of atdata_s[119] is 0. On writes, sets atdata_m[119] to 0 **1** On reads, the value of atdata_s[119] is 1. On writes, sets atdata_m[119] to 1
[14]	`0b0`	ATDATA111	Read-write	Reads atdata_s[111] and writes atdata_m[111]: **0** On reads, the value of atdata_s[111] is 0. On writes, sets atdata_m[111] to 0 **1** On reads, the value of atdata_s[111] is 1. On writes, sets atdata_m[111] to 1
[13]	`0b0`	ATDATA103	Read-write	Reads atdata_s[103] and writes atdata_m[103]: **0** On reads, the value of atdata_s[103] is 0. On writes, sets atdata_m[103] to 0 **1** On reads, the value of atdata_s[103] is 1. On writes, sets atdata_m[103] to 1
[12]	`0b0`	ATDATA95	Read-write	Reads atdata_s[95] and writes atdata_m[95]: **0** On reads, the value of atdata_s[95] is 0. On writes, sets atdata_m[95] to 0 **1** On reads, the value of atdata_s[95] is 1. On writes, sets atdata_m[95] to 1
[11]	`0b0`	ATDATA87	Read-write	Reads atdata_s[87] and writes atdata_m[87]: **0** On reads, the value of atdata_s[87] is 0. On writes, sets atdata_m[87] to 0 **1** On reads, the value of atdata_s[87] is 1. On writes, sets atdata_m[87] to 1
[10]	`0b0`	ATDATA79	Read-write	Reads atdata_s[79] and writes atdata_m[79]: **0** On reads, the value of atdata_s[79] is 0. On writes, sets atdata_m[79] to 0 **1** On reads, the value of atdata_s[79] is 1. On writes, sets atdata_m[79] to 1
[9]	`0b0`	ATDATA71	Read-write	Reads atdata_s[71] and writes atdata_m[71]: **0** On reads, the value of atdata_s[71] is 0. On writes, sets atdata_m[71] to 0 **1** On reads, the value of atdata_s[71] is 1. On writes, sets atdata_m[71] to 1

Bits	Reset value	Name	Type	Function
[8]	`0b0`	ATDATA63	Read-write	Reads atdata_s[63] and writes atdata_m[63]: **0** On reads, the value of atdata_s[63] is 0. On writes, sets atdata_m[63] to 0 **1** On reads, the value of atdata_s[63] is 1. On writes, sets atdata_m[63] to 1
[7]	`0b0`	ATDATA55	Read-write	Reads atdata_s[55] and writes atdata_m[55]: **0** On reads, the value of atdata_s[55] is 0. On writes, sets atdata_m[55] to 0 **1** On reads, the value of atdata_s[55] is 1. On writes, sets atdata_m[55] to 1
[6]	`0b0`	ATDATA47	Read-write	Reads atdata_s[47] and writes atdata_m[47]: **0** On reads, the value of atdata_s[47] is 0. On writes, sets atdata_m[47] to 0 **1** On reads, the value of atdata_s[47] is 1. On writes, sets atdata_m[47] to 1
[5]	`0b0`	ATDATA39	Read-write	Reads atdata_s[39] and writes atdata_m[39]: **0** On reads, the value of atdata_s[39] is 0. On writes, sets atdata_m[39] to 0 **1** On reads, the value of atdata_s[39] is 1. On writes, sets atdata_m[39] to 1
[4]	`0b0`	ATDATA31	Read-write	Reads atdata_s[31] and writes atdata_m[31]: **0** On reads, the value of atdata_s[31] is 0. On writes, sets atdata_m[31] to 0 **1** On reads, the value of atdata_s[31] is 1. On writes, sets atdata_m[31] to 1
[3]	`0b0`	ATDATA23	Read-write	Reads atdata_s[23] and writes atdata_m[23]: **0** On reads, the value of atdata_s[23] is 0. On writes, sets atdata_m[23] to 0 **1** On reads, the value of atdata_s[23] is 1. On writes, sets atdata_m[23] to 1
[2]	`0b0`	ATDATA15	Read-write	Reads atdata_s[15] and writes atdata_m[15]: **0** On reads, the value of atdata_s[15] is 0. On writes, sets atdata_m[15] to 0 **1** On reads, the value of atdata_s[15] is 1. On writes, sets atdata_m[15] to 1
[1]	`0b0`	ATDATA7	Read-write	Reads atdata_s[7] and writes atdata_m[7]: **0** On reads, the value of atdata_s[7] is 0. On writes, sets atdata_m[7] to 0 **1** On reads, the value of atdata_s[7] is 1. On writes, sets atdata_m[7] to 1
[0]	`0b0`	ATDATA0	Read-write	Reads atdata_s[0] and writes atdata_m[0]: **0** On reads, the value of atdata_s[0] is 0. On writes, sets atdata_m[0] to 0 **1** On reads, the value of atdata_s[0] is 1. On writes, sets atdata_m[0] to 1

10.9.2.4 css600_atbfunnel Integration test control register 3, ITATBCTR3

The ITATBCTR3 register enables you to observe and control the SYNCREQ signals into and out of the funnel. Only one slave interface must be selected for integration test. The syncreq receiver on the master interface has a latching function to capture a pulse arriving on that input. The arrival of a pulse sets the latch so that the value can be read. Reading the register clears the latch. Reading a 1 indicates that a syncreq_m pulse arrived since the last read. Reading a 0 indicates that no syncreq_m pulse has arrived. Writing a 1 to the register causes a syncreq_s pulse to be generated to the upstream component.

Attributes

Offset

`0x0EF0`

Type

Read-write

Reset

`0x00000000`

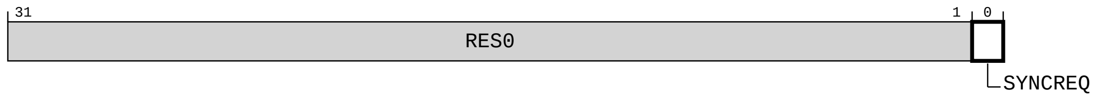
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-192: css600_atbfunnel_ITATBCTR3



The following table shows the bit assignments.

Table 10-200: ITATBCTR3 attributes

Bits	Reset value	Name	Type	Function
[31:1]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[0]	`0b0`	SYNCREQ	Read-write	Reads and controls the SYNCREQ signals into, and out of, the funnel. Reading clears the latch. **0** On reads: no syncreq_m pulse has arrived. On writes: no effect. **1** On reads: a syncreq_m pulse arrived since the last read. On writes: generates a syncreq_s pulse to the upstream component.

10.9.2.5 css600_atbfunnel Integration test control register 2, ITATBCTR2

The ITATBCTR2 register enables you to observe and control the afvalid and atready signals into and out of the funnel. For slave signals coming into the funnel, the register views the ports that are selected through the FUNNELCONTROL register. Only one port must be selected for integration test.

Attributes

Offset

`0x0EF4`

Type

Read-write

Reset

`0x00000000`

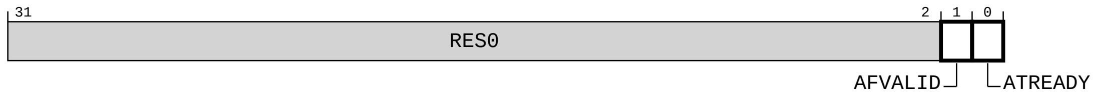
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-193: css600_atbfunnel_ITATBCTR2



The following table shows the bit assignments.

Table 10-201: ITATBCTR2 attributes

Bits	Reset value	Name	Type	Function
[31:2]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[1]	`0b0`	AFVALID	Read-write	Reads and controls the afvalid signals into, and out of, the funnel: **0** On reads: afvalid_m is LOW. On writes: sets afvalid_s LOW. **1** On reads: afvalid_m is HIGH. On writes: sets afvalid_s HIGH.
[0]	`0b0`	ATREADY	Read-write	Reads and controls the atready signal into, and out of, the funnel: **0** On reads: atready_m is LOW. On writes: sets atready_s LOW. **1** On reads: atready_m is HIGH. On writes: sets atready_s HIGH.

10.9.2.6 css600_atbfunnel Integration test control register 1, ITATBCTR1

The ITATBCTR1 register enables you to observe and control the ATID buses into and out of the funnel. For slave signals coming into the funnel, the register views the ports that are selected through the FUNNELCONTROL register. Only one port must be selected for integration test.

Attributes

Offset

`0x0EF8`

Type

Read-write

Reset

`0x00000000`

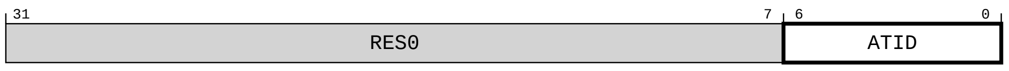
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-194: css600_atbfunnel_ITATBCTR1



The following table shows the bit assignments.

Table 10-202: ITATBCTR1 attributes

Bits	Reset value	Name	Type	Function
[31:7]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[6:0]	`0x0`	ATID	Read-write	When read returns the value on atid_s, when written drives the value on atid_m

10.9.2.7 css600_atbfunnel Integration test control register 0, ITATBCTR0

The ITATBCTR0 register enables you to observabe and control the ATBYTES buses, and the AFREADY and ATVALID signals into and out of the funnel. For slave signals coming into the funnel, the register views the ports that are selected through the FUNNELCONTROL register. Only one port must be selected for integration test.

Attributes

Offset

`0x0EFC`

Type

Read-write

Reset

`0x00000000`

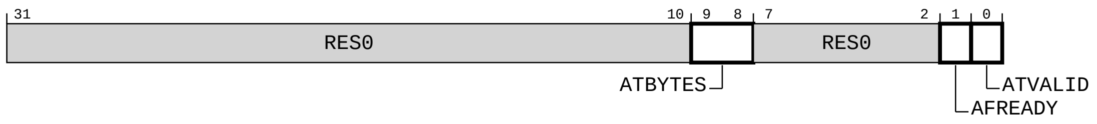
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-195: css600_atbfunnel_ITATBCTR0



The following table shows the bit assignments.

Table 10-203: ITATBCTR0 attributes

Bits	Reset value	Name	Type	Function
[31:10]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[9:8]	`0b00`	ATBYTES	Read-write	Reads the value on atbytes_s[1:0] and writes the values on atbytes_m[1:0]
[7:2]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[1]	`0b0`	AFREADY	Read-write	Reads and controls the afready signals into, and out of, the funnel: **0** On reads: afready_s is LOW. On writes: sets afready_m LOW. **1** On reads: afready_s is HIGH. On writes: sets afready_m HIGH.
[0]	`0b0`	ATVALID	Read-write	Reads and controls the atvalid signals into, and out of, the funnel: **0** On reads: atvalid_s is LOW. On writes: sets atvalid_m LOW. **1** On reads: atvalid_s is HIGH. On writes: sets atvalid_m HIGH.

10.9.2.8 css600_atbfunnel Integration Mode Control Register, ITCTRL

The Integration Mode Control register is used to enable topology detection.

Attributes

Offset

`0x0F00`

Type

Read-write

Reset

`0x00000000`

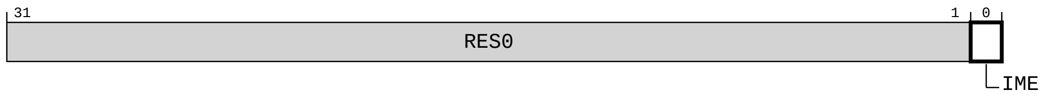
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-196: css600_atbfunnel_ITCTRL



The following table shows the bit assignments.

Table 10-204: ITCTRL attributes

Bits	Reset value	Name	Type	Function
[31:1]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[0]	`0b0`	IME	Read-write	Integration Mode Enable. When set, the component enters integration mode, enabling topology detection or integration testing to be performed.

10.9.2.9 css600_atbfunnel Claim Tag Set Register, CLAIMSET

This register forms one half of the claim tag value. On writes, this location enables individual bits to be set. On reads, it returns the number of bits that can be set.

Attributes

Offset

`0x0FA0`

Type

Read-write

Reset

`0x0000000F`

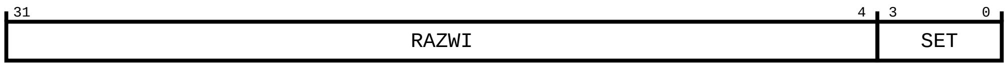
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-197: css600_atbfunnel_CLAIMSET



The following table shows the bit assignments.

Table 10-205: CLAIMSET attributes

Bits	Reset value	Name	Type	Function
[31:4]	`0x0`	RAZ/WI	Read-write	RAZ/WI
[3:0]	`0b1111`	SET	Read-write	A bit-programmable register bank that sets the claim tag value. A read returns a logic 1 for all implemented locations.

10.9.2.10 css600_atbfunnel Claim Tag Clear Register, CLAIMCLR

This register forms one half of the claim tag value. On writes, this location enables individual bits to be cleared. On reads, it returns the current claim tag value.

Attributes

Offset

`0x0FA4`

Type

Read-write

Reset

`0x00000000`

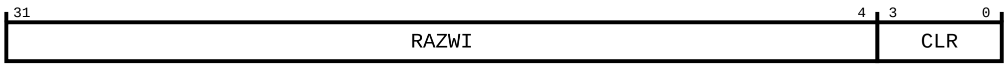
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-198: css600_atbfunnel_CLAIMCLR



The following table shows the bit assignments.

Table 10-206: CLAIMCLR attributes

Bits	Reset value	Name	Type	Function
[31:4]	`0x0`	RAZ/WI	Read-write	RAZ/WI
[3:0]	`0b0000`	CLR	Read-write	A bit-programmable register bank that clears the claim tag value. It is zero at reset. It is used by software agents to signal to each other ownership of the hardware. It has no direct effect on the hardware itself.

10.9.2.11 css600_atbfunnel Device Affinity register 0, DEVAFF0

Enables a debugger to determine if two components have an affinity with each other.

Attributes

Offset

`0x0FA8`

Type

Read-only

Reset

`0x00000000`

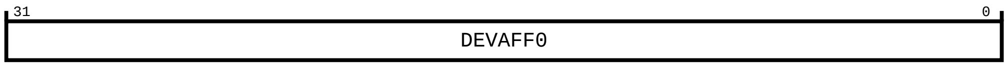
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-199: css600_atbfunnel_DEVAFF0



The following table shows the bit assignments.

Table 10-207: DEVAFF0 attributes

Bits	Reset value	Name	Type	Function
[31:0]	`0x0`	DEVAFF0	Read-only	This field is RAZ.

10.9.2.12 css600_atbfunnel Device Affinity register 1, DEVAFF1

Enables a debugger to determine if two components have an affinity with each other.

Attributes

Offset

`0x0FAC`

Type

Read-only

Reset

`0x00000000`

Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-200: css600_atbfunnel_DEVAFF1

31
0
DEVAFF1

The following table shows the bit assignments.

Table 10-208: DEVAFF1 attributes

Bits	Reset value	Name	Type	Function
[31:0]	`0x0`	DEVAFF1	Read-only	This field is RAZ.

10.9.2.13 css600_atbfunnel Authentication Status Register, AUTHSTATUS

Reports the current status of the authentication control signals.

Attributes

Offset

`0x0FB8`

Type

Read-only

Reset

`0x00000000`

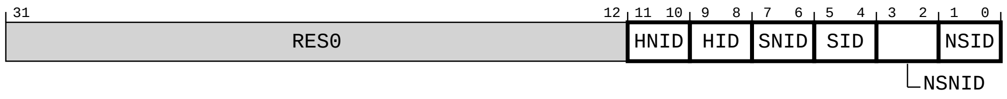
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-201: css600_atbfunnel_AUTHSTATUS



The following table shows the bit assignments.

Table 10-209: AUTHSTATUS attributes

Bits	Reset value	Name	Type	Function
[31:12]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[11:10]	`0b00`	HNID	Read-only	Hypervisor non-invasive debug. **0x0** Functionality not implemented or controlled elsewhere. **0x1** Reserved. **0x2** Functionality disabled. **0x3** Functionality enabled.
[9:8]	`0b00`	HID	Read-only	Hypervisor invasive debug. **0x0** Functionality not implemented or controlled elsewhere. **0x1** Reserved. **0x2** Functionality disabled. **0x3** Functionality enabled.
[7:6]	`0b00`	SNID	Read-only	Secure non-invasive debug. **0x0** Functionality not implemented or controlled elsewhere. **0x1** Reserved. **0x2** Functionality disabled. **0x3** Functionality enabled.
[5:4]	`0b00`	SID	Read-only	Secure invasive debug. **0x0** Functionality not implemented or controlled elsewhere. **0x1** Reserved. **0x2** Functionality disabled. **0x3** Functionality enabled.
[3:2]	`0b00`	NSNID	Read-only	Non-secure non-invasive debug. **0x0** Functionality not implemented or controlled elsewhere. **0x1** Reserved. **0x2** Functionality disabled. **0x3** Functionality enabled.
[1:0]	`0b00`	NSID	Read-only	Non-secure invasive debug. **0x0** Functionality not implemented or controlled elsewhere. **0x1** Reserved. **0x2** Functionality disabled. **0x3** Functionality enabled.

10.9.2.14 css600_atbfunnel Device Architecture Register, DEVARCH

Identifies the architect and architecture of a CoreSight component. The architect might differ from the designer of a component, for example Arm defines the architecture but another company designs and implements the component.

Attributes

Offset

`0x0FBC`

Type

Read-only

Reset

`0x00000000`

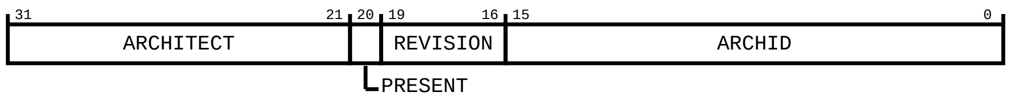
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-202: css600_atbfunnel_DEVARCH



The following table shows the bit assignments.

Table 10-210: DEVARCH attributes

Bits	Reset value	Name	Type	Function
[31:21]	`0x0`	ARCHITECT	Read-only	Returns 0.
[20]	`0b0`	PRESENT	Read-only	Returns 0, indicating that the DEVARCH register is not present.
[19:16]	`0b0000`	REVISION	Read-only	Returns 0
[15:0]	`0x0`	ARCHID	Read-only	Returns 0.

10.9.2.15 css600_atbfunnel Device Configuration Register 2, DEVID2

Contains an **IMPLEMENTATION DEFINED** value.

Attributes

Offset

`0x0FC0`

Type

Read-only

Reset

`0x00000000`

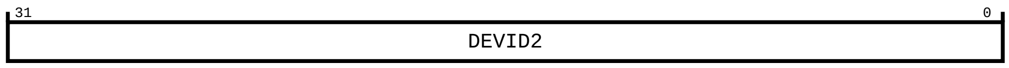
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-203: css600_atbfunnel_DEVID2



The following table shows the bit assignments.

Table 10-211: DEVID2 attributes

Bits	Reset value	Name	Type	Function
[31:0]	`0x0`	DEVID2	Read-only	This field is RAZ.

10.9.2.16 css600_atbfunnel Device Configuration Register 1, DEVID1

This is a device configuration register.

Attributes

Offset

`0x0FC4`

Type

Read-only

Reset

`0x00000000`

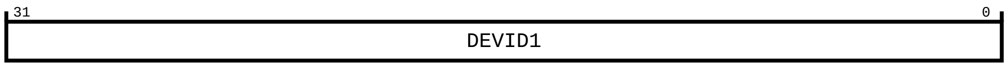
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-204: css600_atbfunnel_DEVID1



The following table shows the bit assignments.

Table 10-212: DEVID1 attributes

Bits	Reset value	Name	Type	Function
[31:0]	`0x0`	DEVID1	Read-only	This field is RAZ

10.9.2.17 css600_atbfunnel Device Configuration Register, DEVID

This register is **IMPLEMENTATION DEFINED** for each Part Number and Designer. The register indicates the capabilities of the component.

Attributes

Offset

`0x0FC8`

Type

Read-only

Reset

`0x0000003-`

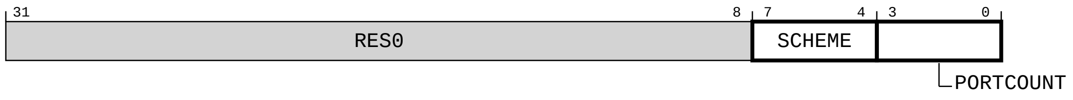
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-205: css600_atbfunnel_DEVID



The following table shows the bit assignments.

Table 10-213: DEVID attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior

Bits	Reset value	Name	Type	Function
[7:4]	`0b0011`	SCHEME	Read-only	Indicates priority scheme implemented. Input priority is controlled by the PRIORITYCONTROL register.
[3:0]	IMPLEMENTATION DEFINED	PORTCOUNT	Read-only	Indicates the number of input ports connected

10.9.2.18 css600_atbfunnel Device Type Identifier Register, DEVTYPE

A debugger can use this register to get information about a component that has an unrecognized Part number.

Attributes

Offset

`0x0FCC`

Type

Read-only

Reset

`0x00000012`

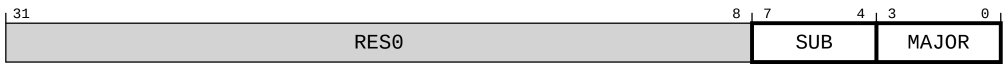
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-206: css600_atbfunnel_DEVTYPE



The following table shows the bit assignments.

Table 10-214: DEVTYPE attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0001`	SUB	Read-only	Minor classification. Returns `0x1`, indicating this component is a Funnel/Router.
[3:0]	`0b0010`	MAJOR	Read-only	Major classification. Returns `0x2`, indicating this component is a Trace Link.

10.9.2.19 css600_atbfunnel Peripheral Identification Register 4, PIDR4

The PIDR4 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FD0`

Type

Read-only

Reset

`0x00000004`

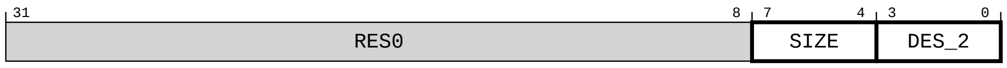
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-207: css600_atbfunnel_PIDR4



The following table shows the bit assignments.

Table 10-215: PIDR4 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0000`	SIZE	Read-only	Indicates the memory size that is used by this component. Returns 0 indicating that the component uses an **UNKNOWN** number of 4KB blocks. Using the SIZE field to indicate the size of the component is deprecated.
[3:0]	`0b0100`	DES_2	Read-only	JEP106 continuation code. Together, with PIDR2.DES_1 and PIDR1.DES_0, they indicate the designer of the component and not the implementer, except where the two are the same.

10.9.2.20 css600_atbfunnel Peripheral Identification Register 5, PIDR5

The PIDR5 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FD4`

Type

Read-only

Reset

`0x00000000`

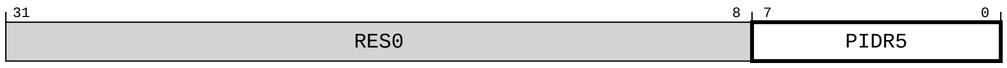
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-208: css600_atbfunnel_PIDR5



The following table shows the bit assignments.

Table 10-216: PIDR5 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	PIDR5	Read-only	Reserved.

10.9.2.21 css600_atbfunnel Peripheral Identification Register 6, PIDR6

The PIDR6 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FD8`

Type

Read-only

Reset

`0x00000000`

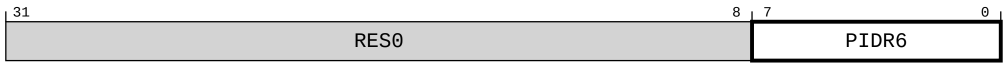
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-209: css600_atbfunnel_PIDR6



The following table shows the bit assignments.

Table 10-217: PIDR6 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	PIDR6	Read-only	Reserved.

10.9.2.22 css600_atbfunnel Peripheral Identification Register 7, PIDR7

The PIDR7 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FDC`

Type

Read-only

Reset

`0x00000000`

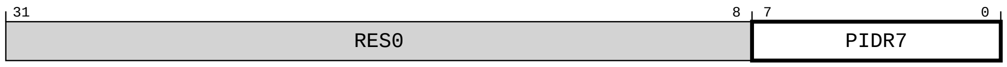
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-210: css600_atbfunnel_PIDR7



The following table shows the bit assignments.

Table 10-218: PIDR7 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	PIDR7	Read-only	Reserved.

10.9.2.23 css600_atbfunnel Peripheral Identification Register 0, PIDR0

The PIDR0 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FE0`

Type

Read-only

Reset

`0x000000EB`

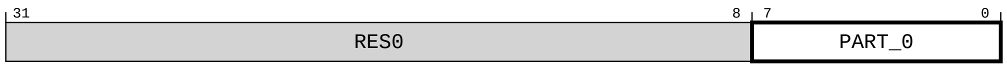
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-211: css600_atbfunnel_PIDR0



The following table shows the bit assignments.

Table 10-219: PIDR0 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0xEB`	PART_0	Read-only	Part number, bits[7:0]. Taken together with PIDR1.PART_1 it indicates the component. The Part Number is selected by the designer of the component.

10.9.2.24 css600_atbfunnel Peripheral Identification Register 1, PIDR1

The PIDR1 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FE4`

Type

Read-only

Reset

`0x000000B9`

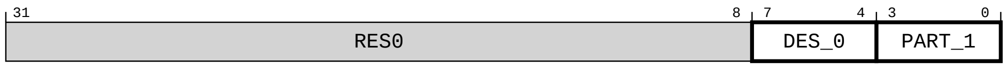
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-212: css600_atbfunnel_PIDR1



The following table shows the bit assignments.

Table 10-220: PIDR1 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b1011`	DES_0	Read-only	JEP106 identification code, bits[3:0]. Together, with PIDR4.DES_2 and PIDR2.DES_1, they indicate the designer of the component and not the implementer, except where the two are the same.
[3:0]	`0b1001`	PART_1	Read-only	Part number, bits[11:8]. Taken together with PIDR0.PART_0 it indicates the component. The Part Number is selected by the designer of the component.

10.9.2.25 css600_atbfunnel Peripheral Identification Register 2, PIDR2

The PIDR2 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FE8`

Type

Read-only

Reset

`0x0000003B`

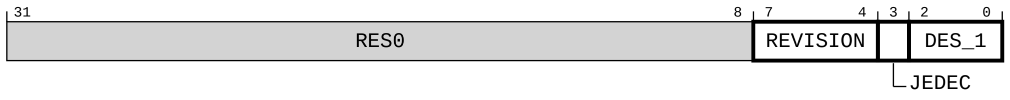
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-213: css600_atbfunnel_PIDR2



The following table shows the bit assignments.

Table 10-221: PIDR2 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0011`	REVISION	Read-only	Revision. It is an incremental value starting at `0x0` for the first design of a component. See the Component list in Chapter 1 for information on the RTL revision of the component.
[3]	`0b1`	JEDEC	Read-only	1 - Always set. Indicates that a JEDEC assigned value is used.
[2:0]	`0b011`	DES_1	Read-only	JEP106 identification code, bits[6:4]. Together, with PIDR4.DES_2 and PIDR1.DES_0, they indicate the designer of the component and not the implementer, except where the two are the same.

10.9.2.26 css600_atbfunnel Peripheral Identification Register 3, PIDR3

The PIDR3 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FEC`

Type

Read-only

Reset

`0x00000000`

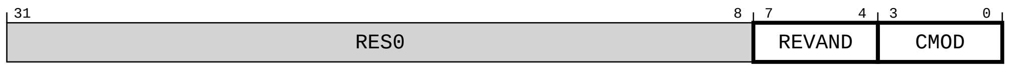
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-214: css600_atbfunnel_PIDR3



The following table shows the bit assignments.

Table 10-222: PIDR3 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0000`	REVAND	Read-only	This field indicates minor errata fixes specific to this design, for example metal fixes after implementation. In most cases this field is `0x0`.
[3:0]	`0b0000`	CMOD	Read-only	Customer Modified. Where the component is reusable IP, this value indicates if the customer has modified the behavior of the component. In most cases this field is `0x0`.

10.9.2.27 css600_atbfunnel Component Identification Register 0, CIDR0

The CIDR0 register is part of the set of component identification registers.

Attributes

Offset

`0x0FF0`

Type

Read-only

Reset

`0x0000000D`

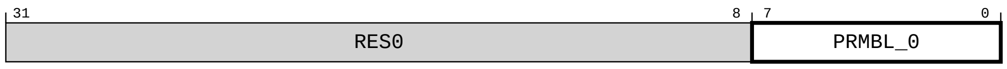
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-215: css600_atbfunnel_CIDR0



The following table shows the bit assignments.

Table 10-223: CIDR0 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0xD`	PRMBL_0	Read-only	Preamble. Returns `0x0D`.

10.9.2.28 css600_atbfunnel Component Identification Register 1, CIDR1

The CIDR1 register is part of the set of component identification registers.

Attributes

Offset

`0x0FF4`

Type

Read-only

Reset

`0x00000090`

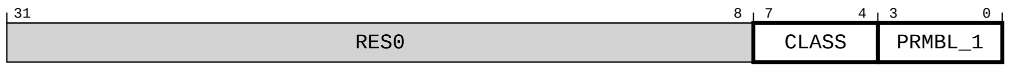
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-216: css600_atbfunnel_CIDR1



The following table shows the bit assignments.

Table 10-224: CIDR1 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b1001`	CLASS	Read-only	Component class. Returns `0x9`, indicating this is a CoreSight component.
[3:0]	`0b0000`	PRMBL_1	Read-only	Preamble. Returns `0x0`.

10.9.2.29 css600_atbfunnel Component Identification Register 2, CIDR2

The CIDR2 register is part of the set of component identification registers.

Attributes

Offset

`0x0FF8`

Type

Read-only

Reset

`0x00000005`

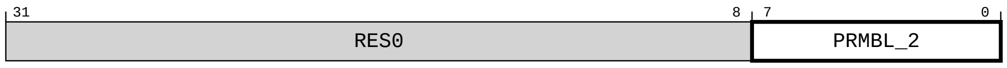
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-217: css600_atbfunnel_CIDR2



The following table shows the bit assignments.

Table 10-225: CIDR2 attributes

Bits	Reset value	Name	Type	Function
[31:8]	0x0	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	0x5	PRMBL_2	Read-only	Preamble. Returns 0x05.

10.9.2.30 css600_atbfunnel Component Identification Register 3, CIDR3

The CIDR3 register is part of the set of component identification registers.

Attributes

Offset

0x0FFC

Type

Read-only

Reset

0x000000B1

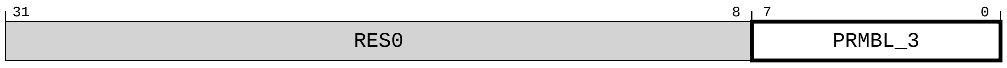
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-218: css600_atbfunnel_CIDR3



The following table shows the bit assignments.

Table 10-226: CIDR3 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0xB1`	PRMBL_3	Read-only	Preamble. Returns `0xB1`.

10.10 `css600_atbreplicator_prog` introduction

This section describes the programmers model of the `css600_atbreplicator_prog`.

10.10.1 css600_atbreplicator register summary

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

Note

A reset value containing one or more '-' means that this register contains **UNKNOWN** or **IMPLEMENTATION-DEFINED** values. See the relevant register description for more information.

Locations that are not listed in the table are Reserved.

Table 10-227: css600_atbreplicator register summary

Offset	Name	Type	Reset	Width	Description
`0x0000`	IDFILT0	RW	`0x00000000`	32	css600_atbreplicator ID filtering control 0 register, IDFILT0
`0x0004`	IDFILT1	RW	`0x00000000`	32	css600_atbreplicator ID filtering control 1 register, IDFILT1
`0x0EF8`	ITATBCTRL	RW	`0x00000000`	32	css600_atbreplicator Integration Test Control register, ITATBCTRL
`0x0EFC`	ITATBSTAT	RW	`0x00000000`	32	css600_atbreplicator Integration Test Status register, ITATBSTAT
`0x0F00`	ITCTRL	RW	`0x00000000`	32	css600_atbreplicator Integration Mode Control Register, ITCTRL
`0x0FA0`	CLAIMSET	RW	`0x0000000F`	32	css600_atbreplicator Claim Tag Set Register, CLAIMSET
`0x0FA4`	CLAIMCLR	RW	`0x00000000`	32	css600_atbreplicator Claim Tag Clear Register, CLAIMCLR
`0x0FA8`	DEVAFF0	RO	`0x00000000`	32	css600_atbreplicator Device Affinity register 0, DEVAFF0
`0x0FAC`	DEVAFF1	RO	`0x00000000`	32	css600_atbreplicator Device Affinity register 1, DEVAFF1
`0x0FB8`	AUTHSTATUS	RO	`0x00000000`	32	css600_atbreplicator Authentication Status Register, AUTHSTATUS
`0x0FBC`	DEVARCH	RO	`0x00000000`	32	css600_atbreplicator Device Architecture Register, DEVARCH
`0x0FC0`	DEVID2	RO	`0x00000000`	32	css600_atbreplicator Device Configuration Register 2, DEVID2
`0x0FC4`	DEVID1	RO	`0x00000000`	32	css600_atbreplicator Device Configuration Register 1, DEVID1
`0x0FC8`	DEVID	RO	`0x00000032`	32	css600_atbreplicator Device Configuration Register, DEVID
`0x0FCC`	DEVTYPE	RO	`0x00000022`	32	css600_atbreplicator Device Type Identifier Register, DEVTYPE
`0x0FD0`	PIDR4	RO	`0x00000004`	32	css600_atbreplicator Peripheral Identification Register 4, PIDR4
`0x0FD4`	PIDR5	RO	`0x00000000`	32	css600_atbreplicator Peripheral Identification Register 5, PIDR5
`0x0FD8`	PIDR6	RO	`0x00000000`	32	css600_atbreplicator Peripheral Identification Register 6, PIDR6
`0x0FDC`	PIDR7	RO	`0x00000000`	32	css600_atbreplicator Peripheral Identification Register 7, PIDR7

Offset	Name	Type	Reset	Width	Description
0x0FE0	PIDR0	RO	0x000000EC	32	css600_atbreplicator Peripheral Identification Register 0, PIDR0
0x0FE4	PIDR1	RO	0x000000B9	32	css600_atbreplicator Peripheral Identification Register 1, PIDR1
0x0FE8	PIDR2	RO	0x0000003B	32	css600_atbreplicator Peripheral Identification Register 2, PIDR2
0x0FEC	PIDR3	RO	0x00000000	32	css600_atbreplicator Peripheral Identification Register 3, PIDR3
0x0FF0	CIDR0	RO	0x0000000D	32	css600_atbreplicator Component Identification Register 0, CIDR0
0x0FF4	CIDR1	RO	0x00000090	32	css600_atbreplicator Component Identification Register 1, CIDR1
0x0FF8	CIDR2	RO	0x00000005	32	css600_atbreplicator Component Identification Register 2, CIDR2
0x0FFC	CIDR3	RO	0x000000B1	32	css600_atbreplicator Component Identification Register 3, CIDR3

10.10.2 Register descriptions

This section describes the `css600_atbreplicator_prog` registers.

css600_atbreplicator register summary provides cross references to individual registers.

10.10.2.1 css600_atbreplicator ID filtering control 0 register, IDFILT0

The IDFILT0 register controls ID filtering for master interface 0.

Attributes

Offset

`0x0000`

Type

Read-write

Reset

`0x00000000`

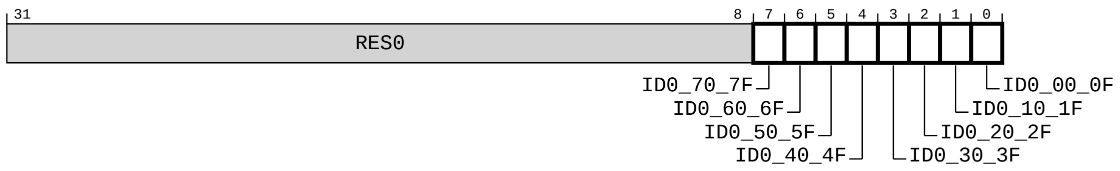
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-219: css600_atbreplicator_IDFILT0



The following table shows the bit assignments.

Table 10-228: IDFILT0 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[7]	`0b0`	ID0_70_7F	Read-write	Enable/disable ID filtering for IDs `0x70` to `0x7F`: **0** Transactions with these IDs are passed on to master interface 0 **1** Transactions with these IDs are discarded by the replicator
[6]	`0b0`	ID0_60_6F	Read-write	Enable/disable ID filtering for IDs `0x60` to `0x6F`: **0** Transactions with these IDs are passed on to master interface 0 **1** Transactions with these IDs are discarded by the replicator
[5]	`0b0`	ID0_50_5F	Read-write	Enable/disable ID filtering for IDs `0x50` to `0x5F`: **0** Transactions with these IDs are passed on to master interface 0 **1** Transactions with these IDs are discarded by the replicator
[4]	`0b0`	ID0_40_4F	Read-write	Enable/disable ID filtering for IDs `0x40` to `0x4F`: **0** Transactions with these IDs are passed on to master interface 0 **1** Transactions with these IDs are discarded by the replicator
[3]	`0b0`	ID0_30_3F	Read-write	Enable/disable ID filtering for IDs `0x30` to `0x3F`: **0** Transactions with these IDs are passed on to master interface 0 **1** Transactions with these IDs are discarded by the replicator
[2]	`0b0`	ID0_20_2F	Read-write	Enable/disable ID filtering for IDs `0x20` to `0x2F`: **0** Transactions with these IDs are passed on to master interface 0 **1** Transactions with these IDs are discarded by the replicator
[1]	`0b0`	ID0_10_1F	Read-write	Enable/disable ID filtering for IDs `0x10` to `0x1F`: **0** Transactions with these IDs are passed on to master interface 0 **1** Transactions with these IDs are discarded by the replicator
[0]	`0b0`	ID0_00_0F	Read-write	Enable/disable ID filtering for IDs `0x00` to `0x0F`: **0** Transactions with these IDs are passed on to master interface 0 **1** Transactions with these IDs are discarded by the replicator

10.10.2.2 css600_atbreplicator ID filtering control 1 register, IDFILT1

The IDFILT1 register controls ID filtering for master interface 1.

Attributes

Offset

`0x0004`

Type

Read-write

Reset

`0x00000000`

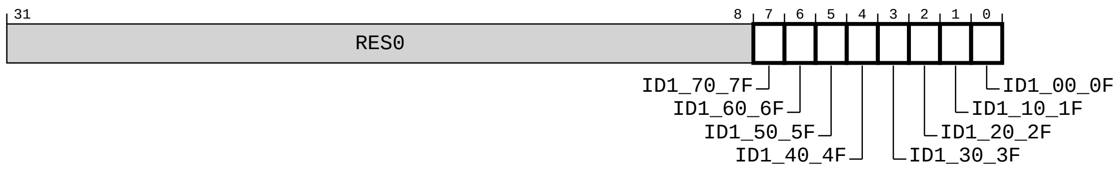
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-220: css600_atbreplicator_IDFILT1



The following table shows the bit assignments.

Table 10-229: IDFILT1 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[7]	`0b0`	ID1_70_7F	Read-write	Enable/disable ID filtering for IDs `0x70` to `0x7F`: **0** Transactions with these IDs are passed on to master interface 1 **1** Transactions with these IDs are discarded by the replicator
[6]	`0b0`	ID1_60_6F	Read-write	Enable/disable ID filtering for IDs `0x60` to `0x6F`: **0** Transactions with these IDs are passed on to master interface 1 **1** Transactions with these IDs are discarded by the replicator
[5]	`0b0`	ID1_50_5F	Read-write	Enable/disable ID filtering for IDs `0x50` to `0x5F`: **0** Transactions with these IDs are passed on to master interface 1 **1** Transactions with these IDs are discarded by the replicator
[4]	`0b0`	ID1_40_4F	Read-write	Enable/disable ID filtering for IDs `0x40` to `0x4F`: **0** Transactions with these IDs are passed on to master interface 1 **1** Transactions with these IDs are discarded by the replicator
[3]	`0b0`	ID1_30_3F	Read-write	Enable/disable ID filtering for IDs `0x30` to `0x3F`: **0** Transactions with these IDs are passed on to master interface 1 **1** Transactions with these IDs are discarded by the replicator
[2]	`0b0`	ID1_20_2F	Read-write	Enable/disable ID filtering for IDs `0x20` to `0x2F`: **0** Transactions with these IDs are passed on to master interface 1 **1** Transactions with these IDs are discarded by the replicator
[1]	`0b0`	ID1_10_1F	Read-write	Enable/disable ID filtering for IDs `0x10` to `0x1F`: **0** Transactions with these IDs are passed on to master interface 1 **1** Transactions with these IDs are discarded by the replicator

Bits	Reset value	Name	Type	Function
[0]	0b0	ID1_00_0F	Read-write	Enable/disable ID filtering for IDs 0x00 to 0x0F: **0** Transactions with these IDs are passed on to master interface 1 **1** Transactions with these IDs are discarded by the replicator

10.10.2.3 css600_atbreplicator Integration Test Control register, ITATBCTRL

The ITATBCTRL register controls atready_s and atvalid_m.

Attributes

Offset

`0x0EF8`

Type

Read-write

Reset

`0x00000000`

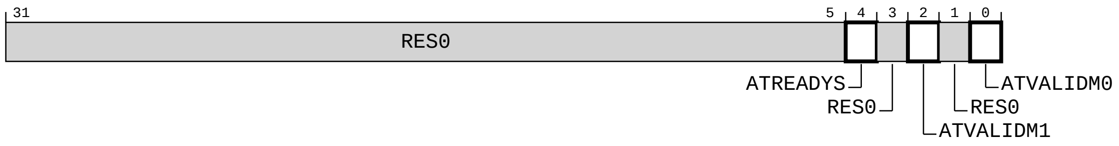
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-221: css600_atbreplicator_ITATBCTRL



The following table shows the bit assignments.

Table 10-230: ITATBCTRL attributes

Bits	Reset value	Name	Type	Function
[31:5]	0x0	RES0	Read-write	Reserved bit or field with SBZP behavior
[4]	0b0	ATREADYS	Read-write	On reads: returns the value written to the register. On writes: **0** Sets static 0 on atready_s **1** Sets static 1 on atready_s
[3]	0b0	RES0	Read-write	Reserved bit or field with SBZP behavior
[2]	0b0	ATVALIDM1	Read-write	On reads: returns the value written to the register. On writes: **0** Sets static 0 on atvalid_m1 **1** Sets static 1 on atvalid_m1

Bits	Reset value	Name	Type	Function
[1]	0b0	RES0	Read-write	Reserved bit or field with SBZP behavior
[0]	0b0	ATVALIDM0	Read-write	On reads: returns the value written to the register. On writes: **0** Sets static 0 on atvalid_m0 **1** Sets static 1 on atvalid_m0

10.10.2.4 css600_atbreplicator Integration Test Status register, ITATBSTAT

The ITATBSTAT register controls atvalid_s and atready_m.

Attributes

Offset

`0x0EFC`

Type

Read-write

Reset

`0x00000000`

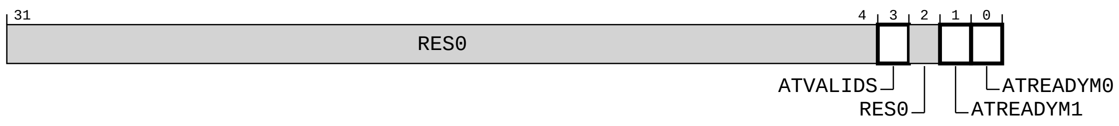
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-222: css600_atbreplicator_ITATBSTAT



The following table shows the bit assignments.

Table 10-231: ITATBSTAT attributes

Bits	Reset value	Name	Type	Function
[31:4]	0x0	RES0	Read-write	Reserved bit or field with SBZP behavior
[3]	0b0	ATVALIDS	Read-only	Returns the value on atvalid_s
[2]	0b0	RES0	Read-write	Reserved bit or field with SBZP behavior
[1]	0b0	ATREADYM1	Read-only	Returns the value on atready_m1
[0]	0b0	ATREADYM0	Read-only	Returns the value on atready_m0

10.10.2.5 css600_atbreplicator Integration Mode Control Register, ITCTRL

The Integration Mode Control register is used to enable topology detection.

Attributes

Offset

`0x0F00`

Type

Read-write

Reset

`0x00000000`

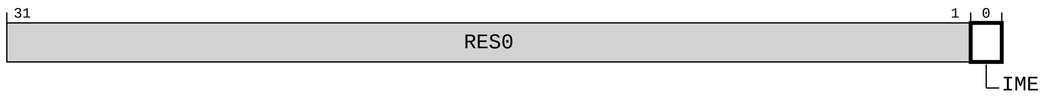
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-223: css600_atbreplicator_ITCTRL



The following table shows the bit assignments.

Table 10-232: ITCTRL attributes

Bits	Reset value	Name	Type	Function
[31:1]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[0]	`0b0`	IME	Read-write	Integration Mode Enable. When set, the component enters integration mode, enabling topology detection or integration testing to be performed.

10.10.2.6 css600_atbreplicator Claim Tag Set Register, CLAIMSET

This register forms one half of the claim tag value. On writes, this location enables individual bits to be set. On reads, it returns the number of bits that can be set.

Attributes

Offset

`0x0FA0`

Type

Read-write

Reset

`0x0000000F`

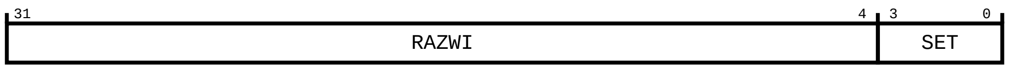
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-224: css600_atbreplicator_CLAIMSET



The following table shows the bit assignments.

Table 10-233: CLAIMSET attributes

Bits	Reset value	Name	Type	Function
[31:4]	`0x0`	RAZ/WI	Read-write	RAZ/WI
[3:0]	`0b1111`	SET	Read-write	A bit-programmable register bank that sets the claim tag value. A read returns a logic 1 for all implemented locations.

10.10.2.7 css600_atbreplicator Claim Tag Clear Register, CLAIMCLR

This register forms one half of the claim tag value. On writes, this location enables individual bits to be cleared. On reads, it returns the current claim tag value.

Attributes

Offset

`0x0FA4`

Type

Read-write

Reset

`0x00000000`

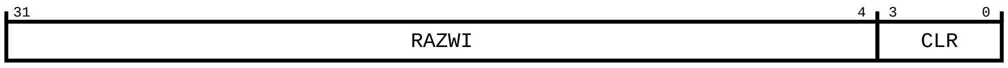
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-225: css600_atbreplicator_CLAIMCLR



The following table shows the bit assignments.

Table 10-234: CLAIMCLR attributes

Bits	Reset value	Name	Type	Function
[31:4]	`0x0`	RAZ/WI	Read-write	RAZ/WI
[3:0]	`0b0000`	CLR	Read-write	A bit-programmable register bank that clears the claim tag value. It is zero at reset. It is used by software agents to signal to each other ownership of the hardware. It has no direct effect on the hardware itself.

10.10.2.8 css600_atbreplicator Device Affinity register 0, DEVAFF0

Enables a debugger to determine if two components have an affinity with each other.

Attributes

Offset

`0x0FA8`

Type

Read-only

Reset

`0x00000000`

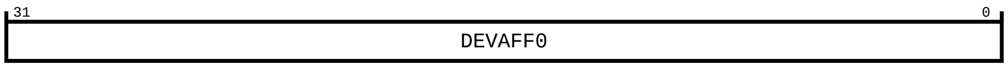
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-226: css600_atbreplicator_DEVAFF0



The following table shows the bit assignments.

Table 10-235: DEVAFF0 attributes

Bits	Reset value	Name	Type	Function
[31:0]	`0x0`	DEVAFF0	Read-only	This field is RAZ.

10.10.2.9 css600_atbreplicator Device Affinity register 1, DEVAFF1

Enables a debugger to determine if two components have an affinity with each other.

Attributes

Offset

`0x0FAC`

Type

Read-only

Reset

`0x00000000`

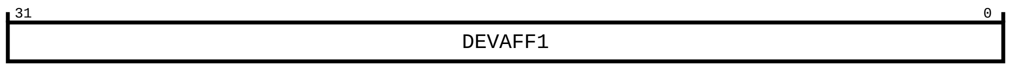
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-227: css600_atbreplicator_DEVAFF1



The following table shows the bit assignments.

Table 10-236: DEVAFF1 attributes

Bits	Reset value	Name	Type	Function
[31:0]	`0x0`	DEVAFF1	Read-only	This field is RAZ.

10.10.2.10 css600_atbreplicator Authentication Status Register, AUTHSTATUS

Reports the current status of the authentication control signals.

Attributes

Offset

`0x0FB8`

Type

Read-only

Reset

`0x00000000`

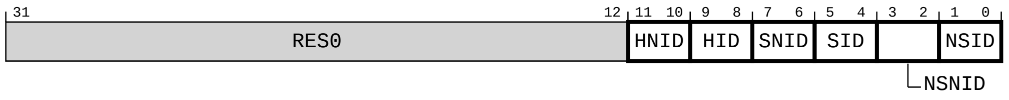
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-228: css600_atbreplicator_AUTHSTATUS



The following table shows the bit assignments.

Table 10-237: AUTHSTATUS attributes

Bits	Reset value	Name	Type	Function
[31:12]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[11:10]	`0b00`	HNID	Read-only	Hypervisor non-invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.
[9:8]	`0b00`	HID	Read-only	Hypervisor invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.
[7:6]	`0b00`	SNID	Read-only	Secure non-invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.
[5:4]	`0b00`	SID	Read-only	Secure invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.
[3:2]	`0b00`	NSNID	Read-only	Non-secure non-invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.

Bits	Reset value	Name	Type	Function
[1:0]	0b00	NSID	Read-only	Non-secure invasive debug. **0x0** Functionality not implemented or controlled elsewhere. **0x1** Reserved. **0x2** Functionality disabled. **0x3** Functionality enabled.

10.10.2.11 css600_atbreplicator Device Architecture Register, DEVARCH

Identifies the architect and architecture of a CoreSight component. The architect might differ from the designer of a component, for example Arm defines the architecture but another company designs and implements the component.

Attributes

Offset

`0x0FBC`

Type

Read-only

Reset

`0x00000000`

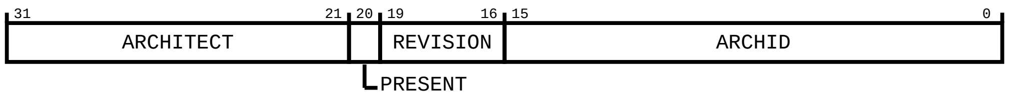
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-229: css600_atbreplicator_DEVARCH



The following table shows the bit assignments.

Table 10-238: DEVARCH attributes

Bits	Reset value	Name	Type	Function
[31:21]	0x0	ARCHITECT	Read-only	Returns 0.
[20]	0b0	PRESENT	Read-only	Returns 0, indicating that the DEVARCH register is not present.
[19:16]	0b0000	REVISION	Read-only	Returns 0
[15:0]	0x0	ARCHID	Read-only	Returns 0.

10.10.2.12 css600_atbreplicator Device Configuration Register 2, DEVID2

Contains an **IMPLEMENTATION DEFINED** value.

Attributes

Offset
: `0x0FC0`

Type
: Read-only

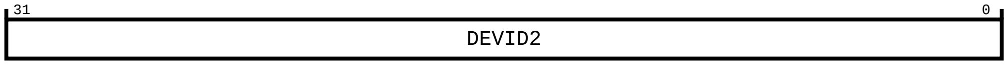
Reset
: `0x00000000`

Width
: 32

Bit descriptions

The following image shows the bit assignments.

Figure 10-230: css600_atbreplicator_DEVID2



The following table shows the bit assignments.

Table 10-239: DEVID2 attributes

Bits	Reset value	Name	Type	Function
[31:0]	`0x0`	DEVID2	Read-only	This field is RAZ.

10.10.2.13 css600_atbreplicator Device Configuration Register 1, DEVID1

This is a device configuration register.

Attributes

Offset
: `0x0FC4`

Type
: Read-only

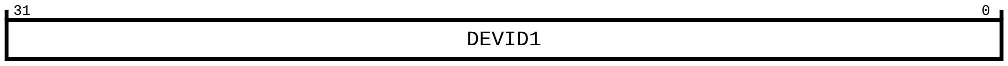
Reset
: `0x00000000`

Width
: 32

Bit descriptions

The following image shows the bit assignments.

Figure 10-231: css600_atbreplicator_DEVID1



The following table shows the bit assignments.

Table 10-240: DEVID1 attributes

Bits	Reset value	Name	Type	Function
[31:0]	0x0	DEVID1	Read-only	This field is RAZ.

10.10.2.14 css600_atbreplicator Device Configuration Register, DEVID

This register is **IMPLEMENTATION DEFINED** for each Part Number and Designer. The register indicates the capabilities of the component.

Attributes

Offset

`0x0FC8`

Type

Read-only

Reset

`0x00000032`

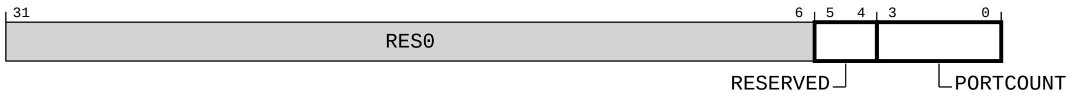
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-232: css600_atbreplicator_DEVID



The following table shows the bit assignments.

Table 10-241: DEVID attributes

Bits	Reset value	Name	Type	Function
[31:6]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[5:4]	`0b11`	RESERVED	Read-only	Reserved. Returns `0x3`. Software must not rely on this value.
[3:0]	`0b0010`	PORTCOUNT	Read-only	Indicates the number of master ports implemented

10.10.2.15 css600_atbreplicator Device Type Identifier Register, DEVTYPE

A debugger can use this register to get information about a component that has an unrecognized Part number.

Attributes

Offset

`0x0FCC`

Type

Read-only

Reset

`0x00000022`

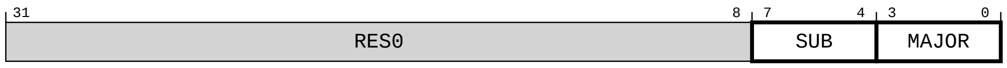
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-233: css600_atbreplicator_DEVTYPE



The following table shows the bit assignments.

Table 10-242: DEVTYPE attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0010`	SUB	Read-only	Minor classification. Returns `0x2`, indicates this component is a Filter.
[3:0]	`0b0010`	MAJOR	Read-only	Major classification. Returns `0x2`, indicating this component is a Trace Link.

10.10.2.16 css600_atbreplicator Peripheral Identification Register 4, PIDR4

The PIDR4 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FD0`

Type

Read-only

Reset

`0x00000004`

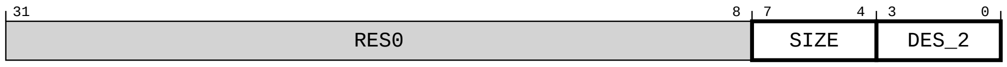
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-234: css600_atbreplicator_PIDR4



The following table shows the bit assignments.

Table 10-243: PIDR4 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0000`	SIZE	Read-only	Indicates the memory size that is used by this component. Returns 0 indicating that the component uses an **UNKNOWN** number of 4KB blocks. Using the SIZE field to indicate the size of the component is deprecated.
[3:0]	`0b0100`	DES_2	Read-only	JEP106 continuation code. Together, with PIDR2.DES_1 and PIDR1.DES_0, they indicate the designer of the component and not the implementer, except where the two are the same.

10.10.2.17 css600_atbreplicator Peripheral Identification Register 5, PIDR5

The PIDR5 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FD4`

Type

Read-only

Reset

`0x00000000`

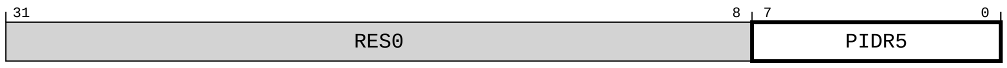
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-235: css600_atbreplicator_PIDR5



The following table shows the bit assignments.

Table 10-244: PIDR5 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	PIDR5	Read-only	Reserved.

10.10.2.18 css600_atbreplicator Peripheral Identification Register 6, PIDR6

The PIDR6 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FD8`

Type

Read-only

Reset

`0x00000000`

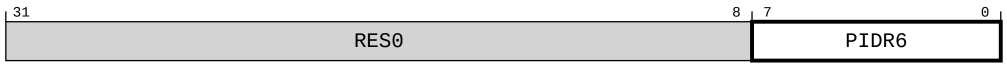
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-236: css600_atbreplicator_PIDR6



The following table shows the bit assignments.

Table 10-245: PIDR6 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	PIDR6	Read-only	Reserved.

10.10.2.19 css600_atbreplicator Peripheral Identification Register 7, PIDR7

The PIDR7 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FDC`

Type

Read-only

Reset

`0x00000000`

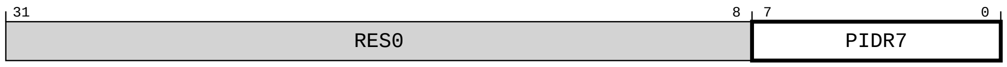
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-237: css600_atbreplicator_PIDR7



The following table shows the bit assignments.

Table 10-246: PIDR7 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	PIDR7	Read-only	Reserved.

10.10.2.20 css600_atbreplicator Peripheral Identification Register 0, PIDR0

The PIDR0 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FE0`

Type

Read-only

Reset

`0x000000EC`

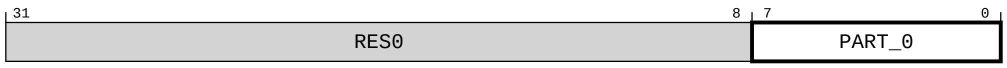
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-238: css600_atbreplicator_PIDR0



The following table shows the bit assignments.

Table 10-247: PIDR0 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0xEC`	PART_0	Read-only	Part number, bits[7:0]. Taken together with PIDR1.PART_1 it indicates the component. The Part Number is selected by the designer of the component.

10.10.2.21 css600_atbreplicator Peripheral Identification Register 1, PIDR1

The PIDR1 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FE4`

Type

Read-only

Reset

`0x000000B9`

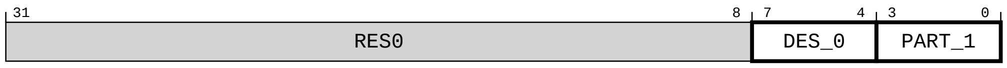
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-239: css600_atbreplicator_PIDR1



The following table shows the bit assignments.

Table 10-248: PIDR1 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b1011`	DES_0	Read-only	JEP106 identification code, bits[3:0]. Together, with PIDR4.DES_2 and PIDR2.DES_1, they indicate the designer of the component and not the implementer, except where the two are the same.
[3:0]	`0b1001`	PART_1	Read-only	Part number, bits[11:8]. Taken together with PIDR0.PART_0 it indicates the component. The Part Number is selected by the designer of the component.

10.10.2.22 css600_atbreplicator Peripheral Identification Register 2, PIDR2

The PIDR2 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FE8`

Type

Read-only

Reset

`0x0000003B`

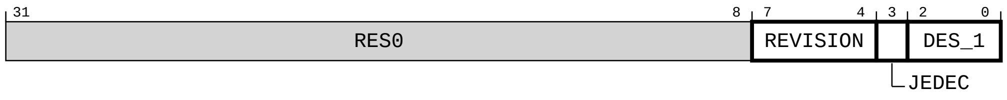
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-240: css600_atbreplicator_PIDR2



The following table shows the bit assignments.

Table 10-249: PIDR2 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0011`	REVISION	Read-only	Revision. It is an incremental value starting at `0x0` for the first design of a component. See the Component list in Chapter 1 for information on the RTL revision of the component.
[3]	`0b1`	JEDEC	Read-only	1 - Always set. Indicates that a JEDEC assigned value is used.
[2:0]	`0b011`	DES_1	Read-only	JEP106 identification code, bits[6:4]. Together, with PIDR4.DES_2 and PIDR1.DES_0, they indicate the designer of the component and not the implementer, except where the two are the same.

10.10.2.23 css600_atbreplicator Peripheral Identification Register 3, PIDR3

The PIDR3 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FEC`

Type

Read-only

Reset

`0x00000000`

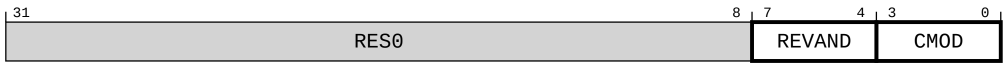
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-241: css600_atbreplicator_PIDR3



The following table shows the bit assignments.

Table 10-250: PIDR3 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0000`	REVAND	Read-only	This field indicates minor errata fixes specific to this design, for example metal fixes after implementation. In most cases this field is `0x0`.
[3:0]	`0b0000`	CMOD	Read-only	Customer Modified. Where the component is reusable IP, this value indicates if the customer has modified the behavior of the component. In most cases this field is `0x0`.

10.10.2.24 css600_atbreplicator Component Identification Register 0, CIDR0

The CIDR0 register is part of the set of component identification registers.

Attributes

Offset

`0x0FF0`

Type

Read-only

Reset

`0x0000000D`

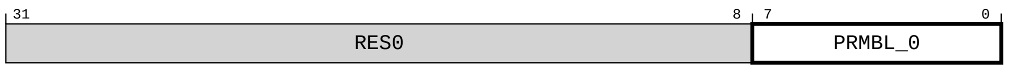
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-242: css600_atbreplicator_CIDR0



The following table shows the bit assignments.

Table 10-251: CIDR0 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0xD`	PRMBL_0	Read-only	Preamble. Returns `0x0D`.

10.10.2.25 css600_atbreplicator Component Identification Register 1, CIDR1

The CIDR1 register is part of the set of component identification registers.

Attributes

Offset

`0x0FF4`

Type

Read-only

Reset

`0x00000090`

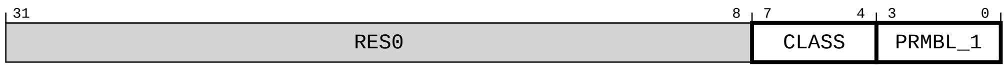
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-243: css600_atbreplicator_CIDR1



The following table shows the bit assignments.

Table 10-252: CIDR1 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b1001`	CLASS	Read-only	Component class. Returns `0x9`, indicating this is a CoreSight component.
[3:0]	`0b0000`	PRMBL_1	Read-only	Preamble. Returns `0x0`.

10.10.2.26 css600_atbreplicator Component Identification Register 2, CIDR2

The CIDR2 register is part of the set of component identification registers.

Attributes

Offset

`0x0FF8`

Type

Read-only

Reset

`0x00000005`

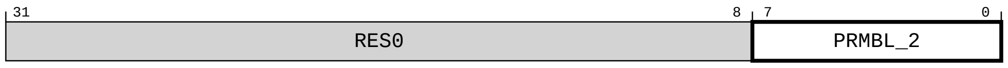
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-244: css600_atbreplicator_CIDR2



The following table shows the bit assignments.

Table 10-253: CIDR2 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x5`	PRMBL_2	Read-only	Preamble. Returns `0x05`.

10.10.2.27 css600_atbreplicator Component Identification Register 3, CIDR3

The CIDR3 register is part of the set of component identification registers.

Attributes

Offset

`0x0FFC`

Type

Read-only

Reset

`0x000000B1`

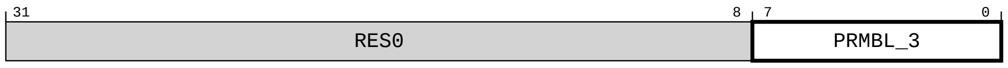
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-245: css600_atbreplicator_CIDR3



The following table shows the bit assignments.

Table 10-254: CIDR3 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0xB1`	PRMBL_3	Read-only	Preamble. Returns `0xB1`.

10.11 `css600_tmc_etb` introduction

This section describes the programmers model of the `css600_tmc_etb`.

10.11.1 css600_tmc_etb register summary

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

Note

A reset value containing one or more '-' means that this register contains **UNKNOWN** or **IMPLEMENTATION-DEFINED** values. See the relevant register description for more information.

Locations that are not listed in the table are Reserved.

Table 10-255: css600_tmc_etb register summary

Offset	Name	Type	Reset	Width	Description
`0x0004`	RSZ	RO	`0x--------`	32	css600_tmc_etb RAM Size register, RSZ
`0x000C`	STS	RO	`0x0000001-`	32	css600_tmc_etb Status register, STS
`0x0010`	RRD	RO	`0x--------`	32	css600_tmc_etb RAM Read Data register, RRD
`0x0014`	RRP	RW	`0x--------`	32	css600_tmc_etb RAM Read Pointer register, RRP
`0x0018`	RWP	RW	`0x--------`	32	css600_tmc_etb RAM Write Pointer register, RWP
`0x001C`	TRG	RW	`0x--------`	32	css600_tmc_etb Trigger Counter register, TRG
`0x0020`	CTL	RW	`0x00000000`	32	css600_tmc_etb Control Register, CTL
`0x0024`	RWD	WO	`0x00000000`	32	css600_tmc_etb RAM Write Data register, RWD
`0x0028`	MODE	RW	`0x000000--`	32	css600_tmc_etb Mode register, MODE
`0x002C`	LBUFLEVEL	RO	`0x--------`	32	css600_tmc_etb Latched Buffer Fill Level, LBUFLEVEL
`0x0030`	CBUFLEVEL	RO	`0x--------`	32	css600_tmc_etb Current Buffer Fill Level, CBUFLEVEL
`0x0034`	BUFWM	RW	`0x--------`	32	css600_tmc_etb Buffer Level Water Mark, BUFWM
`0x0300`	FFSR	RO	`0x0000000-`	32	css600_tmc_etb Formatter and Flush Status Register, FFSR
`0x0304`	FFCR	RW	`0x00000000`	32	css600_tmc_etb Formatter and Flush Control Register, FFCR
`0x0308`	PSCR	RW	`0x0000000A`	32	css600_tmc_etb Periodic Synchronization Counter Register, PSCR
`0x0EE0`	ITEVTINTR	WO	`0x00000000`	32	css600_tmc_etb Integration Test Event and Interrupt Control Register, ITEVTINTR
`0x0EE8`	ITTRFLIN	RO	`0x00000000`	32	css600_tmc_etb Integration Test Trigger In and Flush In register, ITTRFLIN
`0x0EEC`	ITATBDATA0	RO	`0x00000000`	32	css600_tmc_etb Integration Test ATB Data 0 Register, ITATBDATA0
`0x0EF0`	ITATBCTR2	WO	`0x00000000`	32	css600_tmc_etb Integration Test ATB Control 2 Register, ITATBCTR2

Offset	Name	Type	Reset	Width	Description
0x0EF4	ITATBCTR1	RO	0x00000000	32	css600_tmc_etb Integration Test ATB Control 1 Register, ITATBCTR1
0x0EF8	ITATBCTR0	RO	0x00000000	32	css600_tmc_etb Integration Test ATB Control 0 Register, ITATBCTR0
0x0F00	ITCTRL	RW	0x00000000	32	css600_tmc_etb Integration Mode Control Register, ITCTRL
0x0FA0	CLAIMSET	RW	0x0000000F	32	css600_tmc_etb Claim Tag Set Register, CLAIMSET
0x0FA4	CLAIMCLR	RW	0x00000000	32	css600_tmc_etb Claim Tag Clear Register, CLAIMCLR
0x0FB8	AUTHSTATUS	RO	0x00000000	32	css600_tmc_etb Authentication Status Register, AUTHSTATUS
0x0FC4	DEVID1	RO	0x00000001	32	css600_tmc_etb Device Configuration Register 1, DEVID1
0x0FC8	DEVID	RO	0x00000-00	32	css600_tmc_etb Device Configuration Register, DEVID
0x0FCC	DEVTYPE	RO	0x00000021	32	css600_tmc_etb Device Type Identifier Register, DEVTYPE
0x0FD0	PIDR4	RO	0x00000004	32	css600_tmc_etb Peripheral Identification Register 4, PIDR4
0x0FD4	PIDR5	RO	0x00000000	32	css600_tmc_etb Peripheral Identification Register 5, PIDR5
0x0FD8	PIDR6	RO	0x00000000	32	css600_tmc_etb Peripheral Identification Register 6, PIDR6
0x0FDC	PIDR7	RO	0x00000000	32	css600_tmc_etb Peripheral Identification Register 7, PIDR7
0x0FE0	PIDR0	RO	0x000000E9	32	css600_tmc_etb Peripheral Identification Register 0, PIDR0
0x0FE4	PIDR1	RO	0x000000B9	32	css600_tmc_etb Peripheral Identification Register 1, PIDR1
0x0FE8	PIDR2	RO	0x0000006B	32	css600_tmc_etb Peripheral Identification Register 2, PIDR2
0x0FEC	PIDR3	RO	0x00000000	32	css600_tmc_etb Peripheral Identification Register 3, PIDR3
0x0FF0	CIDR0	RO	0x0000000D	32	css600_tmc_etb Component Identification Register 0, CIDR0
0x0FF4	CIDR1	RO	0x00000090	32	css600_tmc_etb Component Identification Register 1, CIDR1
0x0FF8	CIDR2	RO	0x00000005	32	css600_tmc_etb Component Identification Register 2, CIDR2
0x0FFC	CIDR3	RO	0x000000B1	32	css600_tmc_etb Component Identification Register 3, CIDR3

10.11.2 Register descriptions

This section describes the `css600_tmc_etb` registers.

css600_tmc_etb register summary provides cross references to individual registers.

10.11.2.1 css600_tmc_etb RAM Size register, RSZ

The RSZ register defines the size of trace memory in units of 32-bit words.

Attributes

Offset

`0x0004`

Type

Read-only

Reset

`0x--------`

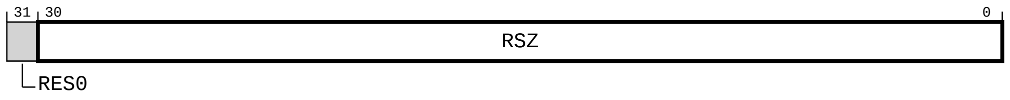
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-246: css600_tmc_etb_RSZ



The following table shows the bit assignments.

Table 10-256: RSZ attributes

Bits	Reset value	Name	Type	Function
[31]	`0b0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[30:0]	IMPLEMENTATION DEFINED	RSZ	Read-only	RAM size. Indicates the size of the RAM in 32-bit words. For example: Returns `0x00000100` if trace memory size is 1KB, `0x40000000` if trace memory size is 4GB. This field has the same value as the MEM_SIZE parameter.

10.11.2.2 css600_tmc_etb Status register, STS

The STS register indicates the status of the Trace Memory Controller. After a reset, software must ignore all the fields of this register except STS.TMCReady. The other fields only have meaning when the TMC has left the Disabled state. Writes to all RO fields of this register are ignored.

Attributes

Offset

`0x000C`

Type

Read-only

Reset

`0x0000001-`

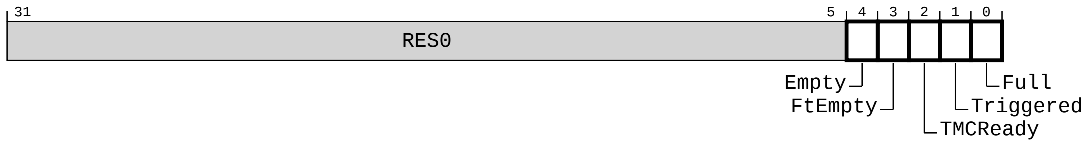
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-247: css600_tmc_etb_STS



The following table shows the bit assignments.

Table 10-257: STS attributes

Bits	Reset value	Name	Type	Function
[31:5]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[4]	`0b1`	Empty	Read-only	Trace buffer empty. If set, this bit indicates that the trace memory does not contain any valid trace data. The reset value of this bit is 1. On leaving Disabled state, this bit dynamically indicates the empty status of trace memory, CBUFLEVEL == 0. However, this does not mean that the pipeline stages within the TMC are empty. To determine whether the internal pipeline stages are empty, the software must read the STS.TMCReady bit. In Circular Buffer mode the Empty bit and the Full bit in this register can be 1 at the same time. This happens because the Full bit in this mode, when set, does not clear until TraceCaptEn is set.
[3]	`0b1`	FtEmpty	Read-only	Trace capture has been completed and all captured trace data has been written to the trace memory, set in Stopped or Disabled state. Otherwise, it is cleared. The reset value is 1
[2]	`0b1`	TMCReady	Read-only	Trace capture has been completed and all captured trace data has been written to the trace memory
[1]	UNKNOWN	Triggered	Read-only	TMC triggered. This bit is set when trace capture is in progress and the TMC has detected a trigger event. This bit is cleared to 0 when leaving the Disabled state and retains its value when entering the Disabled state. A trigger event is when the TMC has written a set number of data words, as programmed in the TRG register, into the trace memory after a rising edge of trigin input, or a trigger packet (atid_s = `0x7D`) is received in the input trace.
[0]	UNKNOWN	Full	Read-only	Trace memory full. This bit helps determine the amount of valid data present in the trace memory. It is not affected by the reprogramming of pointer registers in Disabled state. It is cleared when the TMC leaves Disabled state. In Circular Buffer mode, this flag is set when the RAM write pointer wraps around the top of the buffer. It remains set until the TraceCaptEn bit is cleared and set. In Software FIFO mode, this flag indicates that the current space in the trace memory is less than or equal to the value programmed in the BUFWM Register, that is, Fill level >= MEM_SIZEBUFWM. The FULL output from the TMC reflects the value of this register bit, except when the Integration Mode bit in the ITCTRL Register, `0xF00`, is set.

10.11.2.3 css600_tmc_etb RAM Read Data register, RRD

Reading this register allows data to be read from the trace memory at the location pointed to by the RRP register when either in the Disabled state or operating in CB or SWF1 mode. When ATB_DATA_WIDTH is 32, 64 or 128 bit wide the memory width is twice as wide and a memory word holds 8, 16, or 32 bytes. Multiple RRD reads must be performed to read a full memory word. When a full memory width of data has been read via the RRD register, the RRP register is incremented to the next memory word. When the TMC left the Disabled state and the trace memory is empty, this register returns `0xFFFFFFFF`. When the TMC left the Disabled state and the trace memory is empty, this register returns `0xFFFFFFFF`. When operating in CB mode and the TMC left the Disabled state, this register returns `0xFFFFFFFF` in all other states except the STOPPED state.

Attributes

Offset

`0x0010`

Type

Read-only

Reset

`0x--------`

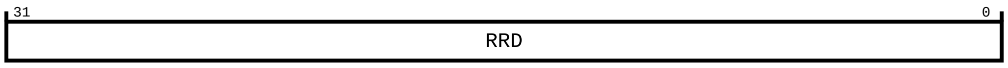
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-248: css600_tmc_etb_RRD



The following table shows the bit assignments.

Table 10-258: RRD attributes

Bits	Reset value	Name	Type	Function
[31:0]	UNKNOWN	RRD	Read-only	Returns the data read from trace memory

10.11.2.4 css600_tmc_etb RAM Read Pointer register, RRP

The RRP register contains the value of the read pointer that is used to read entries from trace memory over the APB interface. Software must program it with the same value as RWP before enabling trace capture.

When the TMC is in the Running or Stopping state the data after formatting needs to be written to memory:

- In CB mode, RRP does not increment initially. When RWP increments to the same value as RRP, then RRP increments too.
- In SWF1 mode, RWP increments until RWP increments to the same value as RRP. Then, writing to memory is stalled. Writing to memory resumes when RRP advances.

When the TMC is in the Running or Stopping state:

- In CB mode, reading from RRD is disabled and RRP is not affected by this.

- In SWF1 mode, RRP advances by reading the data from memory via RRD unless STS.Empty is set (or RRD returns `0xFFFFFFFF`).

When the TMC is in the Stopped state:

- In CB or SWF1 mode, RRP advances by reading the data from memory via RRD unless STS.Empty is set (or RRD returns `0xFFFFFFFF`).

When the TMC is in the Disabled state:

- Writing to RRP is enabled. For all other states writing to RRP is disabled.
- RRP advances by reading the data from memory via RRD.

When the TMC is in the Draining or Disabling state:

- Reading RRP returns unknown value.

Attributes

Offset

`0x0014`

Type

Read-write

Reset

`0x--------`

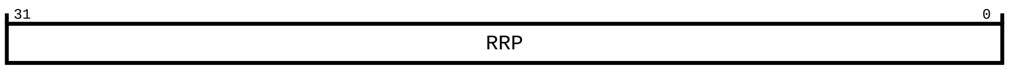
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-249: css600_tmc_etb_RRP



The following table shows the bit assignments.

Table 10-259: RRP attributes

Bits	Reset value	Name	Type	Function
[31:0]	UNKNOWN	RRP	Read-write	The RRP width depends on the size of trace memory and is given by log2(MEM_SIZE x 4). The remaining MSBs of the 32-bit register are of type RAZ/WI. When ATB_DATA_WIDTH is 32, 64 or 128 bit wide the memory width is twice as wide and a memory word holds 8, 16 or 32 bytes. When a full memory width of data has been read via the RRD register, the RRP register is incremented to the next memory word. The lowest 4 bits have access type RAZ/WI when ATB_DATA_WIDTH is 32 or 64 bits. The lowest 5 bits have access type RAZ/WI when ATB_DATA_WIDTH is 128 bits.

10.11.2.5 css600_tmc_etb RAM Write Pointer register, RWP

The RWP register sets the write pointer that writes entries into the trace memory. Software must program it before enabling trace capture.

Attributes

Offset

`0x0018`

Type

Read-write

Reset

`0x--------`

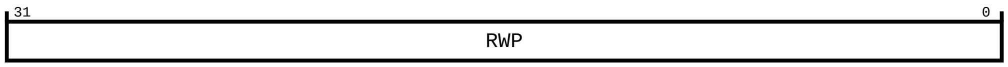
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-250: css600_tmc_etb_RWP



The following table shows the bit assignments.

Table 10-260: RWP attributes

Bits	Reset value	Name	Type	Function
[31:0]	UNKNOWN	RWP	Read-write	The RWP width depends on the size of trace memory and is given by log2(MEM_SIZE x 4). The remaining MSBs of the 32-bit register are of type RAZ/WI. When ATB_DATA_WIDTH is 32, 64 or 128 bit wide the memory width is twice as wide and a memory word holds 8, 16 or 32 bytes. When a full memory width of data has been written to the RWD register, the RWP register is incremented to the next memory word. The lowest 4 bits have access type RAZ/WI when ATB_DATA_WIDTH is 32 or 64 bits. The lowest 5 bits have access type RAZ/WI when ATB_DATA_WIDTH is 128 bits.

10.11.2.6 css600_tmc_etb Trigger Counter register, TRG

The TRG register, in Circular Buffer mode, specifies the number of 32-bit words to capture in the trace memory, after detecting either a rising edge on the trigin input or a trigger packet in the incoming trace stream, that is, where atid_s = `0x7D`. The value programmed must be aligned to the frame length of 128 bits. Software must program this register before leaving Disabled state.

Attributes

Offset

`0x001C`

Type

Read-write

Reset

`0x--------`

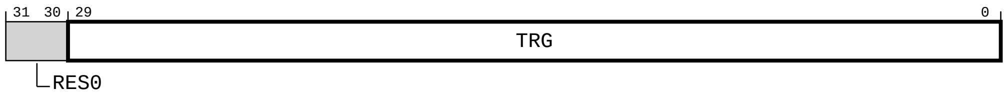
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-251: css600_tmc_etb_TRG



The following table shows the bit assignments.

Table 10-261: TRG attributes

Bits	Reset value	Name	Type	Function
[31:30]	`0b00`	RES0	Read-write	Reserved bit or field with SBZP behavior
[29:0]	UNKNOWN	TRG	Read-write	Trigger count. This count represents the number of 32-bit words of trace that are captured between a trigger packet and a trigger event. The lowest two bits have access type RAZ/WI.

10.11.2.7 css600_tmc_etb Control Register, CTL

The CTL register controls trace stream capture. Setting the CTL.TraceCaptEn bit to 1 enables the TMC to capture the trace data. When trace capture is enabled, the formatter behavior is controlled by the FFCR register.

Attributes

Offset

`0x0020`

Type

Read-write

Reset

`0x00000000`

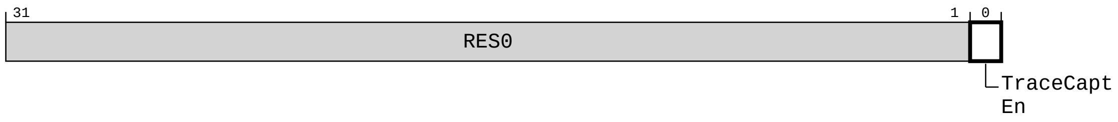
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-252: css600_tmc_etb_CTL



The following table shows the bit assignments.

Table 10-262: CTL attributes

Bits	Reset value	Name	Type	Function
[31:1]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[0]	`0b0`	TraceCaptEn	Read-write	Trace capture enable: **0** Disable trace capture **1** Enable trace capture

10.11.2.8 css600_tmc_etb RAM Write Data register, RWD

The RWD register enables testing of trace memory connectivity to the TMC. Writing this register allows data to be written to the trace memory at the location pointed to by the RWP register when in the Disabled state. When ATB_DATA_WIDTH is 32, 64 or 128 bit wide the memory width is twice as wide and a memory word holds 8, 16 or 32 bytes. Multiple RWD writes must be performed to write a full memory word. When a full memory width of data has been written via the RWD register, the data is written to the trace memory and the RWP register is incremented to the next memory word.

Attributes

Offset

`0x0024`

Type

Write-only

Reset

`0x00000000`

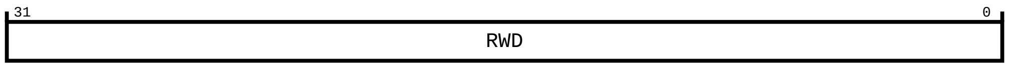
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-253: css600_tmc_etb_RWD



The following table shows the bit assignments.

Table 10-263: RWD attributes

Bits	Reset value	Name	Type	Function
[31:0]	`0x0`	RWD	Write-only	Data written to this register is placed in the trace memory.

10.11.2.9 css600_tmc_etb Mode register, MODE

The MODE register controls the TMC operating mode. The operating mode can only be changed when the TMC is in Disabled state. Attempting to write to this register in any other state results in **UNPREDICTABLE** behavior. The operating mode is ignored when in Disabled state.

Attributes

Offset

`0x0028`

Type

Read-write

Reset

`0x000000--`

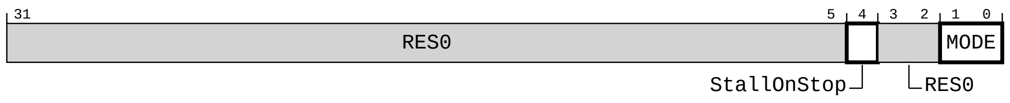
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-254: css600_tmc_etb_MODE



The following table shows the bit assignments.

Table 10-264: MODE attributes

Bits	Reset value	Name	Type	Function
[31:5]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[4]	UNKNOWN	StallOnStop	Read-write	Stall On Stop. If this bit is set and the formatter stops as a result of a stop event, the output atready_s is de-asserted to stall the ATB interface and avoid loss of trace. If this bit is clear and the formatter stops as a result of a stop event, signal atready_s remains asserted but the TMC discards further incoming trace.
[3:2]	`0b00`	RES0	Read-write	Reserved bit or field with SBZP behavior
[1:0]	UNKNOWN	MODE	Read-write	Selects the operating mode after leaving Disabled state. If a reserved MODE value is programmed and trace capture is enabled, the TMC starts to operate in SWF1 mode. However, reading the MODE.MODE field returns the programmed value. **`0x0`** CB, Circular Buffer mode **`0x1`** SWF1, Software Read FIFO mode 1 **`0x2`** Reserved (SWF1) **`0x3`** Reserved (SWF1)

10.11.2.10 css600_tmc_etb Latched Buffer Fill Level, LBUFLEVEL

Reading the LBUFLEVEL register returns the maximum fill level of the trace memory in 32-bit words since this register was last read. Reading this register also results in its contents being updated to the current fill level. When entering Disabled state, it retains its last value. While in Disabled state, reads from this register do not affect its value. When exiting Disabled state, the LBUFLEVEL register is updated to the current fill level.

Attributes

Offset

`0x002C`

Type

Read-only

Reset

`0x--------`

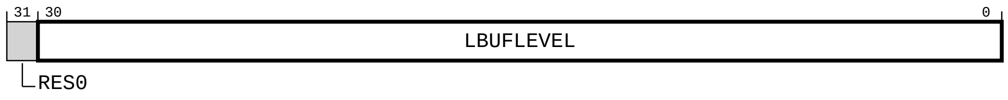
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-255: css600_tmc_etb_LBUFLEVEL



The following table shows the bit assignments.

Table 10-265: LBUFLEVEL attributes

Bits	Reset value	Name	Type	Function
[31]	`0b0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[30:0]	**UNKNOWN**	LBUFLEVEL	Read-only	Latched Buffer Fill Level. Indicates the maximum fill level of the trace memory in 32-bit words since this register was last read. The width of the register is 1 + log2(MEM_SIZE).

10.11.2.11 css600_tmc_etb Current Buffer Fill Level, CBUFLEVEL

The CBUFLEVEL register indicates the current fill level of the trace memory in units of 32-bit words. When the TMC leaves Disabled state, this register dynamically indicates the current fill level of trace memory. It retains its value on entering Disabled state. It is not affected by the reprogramming of pointer registers in Disabled state with the exception of RRD reads and RWD writes. Before leaving the Disabled state software must program RRP with the same value as RWP. Without doing this results in **UNPREDICTABLE** behavior.

Attributes

Offset

`0x0030`

Type

Read-only

Reset

`0x--------`

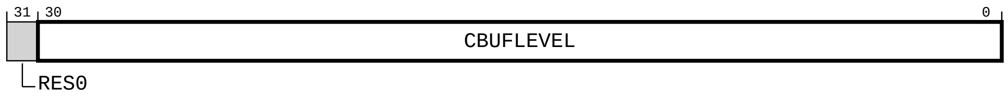
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-256: css600_tmc_etb_CBUFLEVEL



The following table shows the bit assignments.

Table 10-266: CBUFLEVEL attributes

Bits	Reset value	Name	Type	Function
[31]	`0b0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[30:0]	UNKNOWN	CBUFLEVEL	Read-only	Current Buffer Fill Level. Indicates the current fill level of the trace memory in 32-bit words. The width of the register is 1 + log2(MEM_SIZE).

10.11.2.12 css600_tmc_etb Buffer Level Water Mark, BUFWM

The value that is programmed into the BUFWM register indicates the required threshold vacancy level in 32-bit words in the trace memory. When the available space in the FIFO is less than or equal to this value, that is, fill level >= (MEM_SIZE - BUFWM), the full output is asserted and the STS.Full bit is set. This register is used only in the FIFO modes, that is, SWF1, SWF2, and HWF modes. In CB mode, the same functionality is obtained by programming the RWP to the required vacancy trigger level, so that when the pointer wraps around, the full output gets asserted indicating that the vacancy level has fallen below the required level. Reading this register returns the programmed value. The maximum value that can be written into this register is MEM_SIZE - 1, in which case the full output is asserted after the first 32-bit word is written to trace memory. Writing to this register other than when in Disabled state results in **UNPREDICTABLE** behavior. Any software using it must program it with an initial value before setting the CTL.TraceCaptEn bit to 1.

Attributes

Offset

`0x0034`

Type

Read-write

Reset

`0x--------`

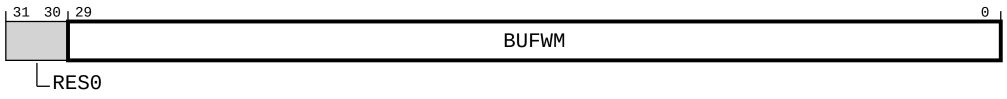
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-257: css600_tmc_etb_BUFWM



The following table shows the bit assignments.

Table 10-267: BUFWM attributes

Bits	Reset value	Name	Type	Function
[31:30]	`0b00`	RES0	Read-write	Reserved bit or field with SBZP behavior
[29:0]	UNKNOWN	BUFWM	Read-write	Buffer Level Watermark. Indicates the desired threshold vacancy level in 32-bit words in the trace memory. The width of the register is log2(MEM_SIZE).

10.11.2.13 css600_tmc_etb Formatter and Flush Status Register, FFSR

This register indicates the status of the Formatter, and the status of Flush request.

Attributes

Offset

`0x0300`

Type

Read-only

Reset

`0x0000000-`

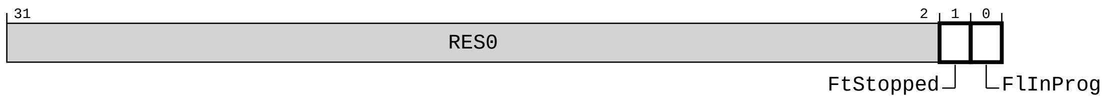
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-258: css600_tmc_etb_FFSR



The following table shows the bit assignments.

Table 10-268: FFSR attributes

Bits	Reset value	Name	Type	Function
[31:2]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[1]	`0b1`	FtStopped	Read-only	Formatter Stopped. This bit behaves the same way as STS.FtEmpty.

Bits	Reset value	Name	Type	Function
[0]	UNKNOWN	FlInProg	Read-only	Flush In Progress. This bit indicates whether the TMC is currently processing a flush request. The flush initiation is controlled by the flush control bits in the FFCR register. This bit is cleared to 0 when leaving the Disabled state and retains its value when entering the Disabled state. When in Disabled state, this bit is not updated. **0** No flush activity in progress. **1** Flush in progress on the ATB slave interface or the TMC internal pipeline.

10.11.2.14 css600_tmc_etb Formatter and Flush Control Register, FFCR

The FFCR controls the generation of stop, trigger and flush events. The insertion of a flush completion packet and the insertion of a trigger packet in the formatted trace is enabled here. Also one of the 2 formatter modes for bypass mode and normal mode can be changed here when the formatter has stopped.

Attributes

Offset

`0x0304`

Type

Read-write

Reset

`0x00000000`

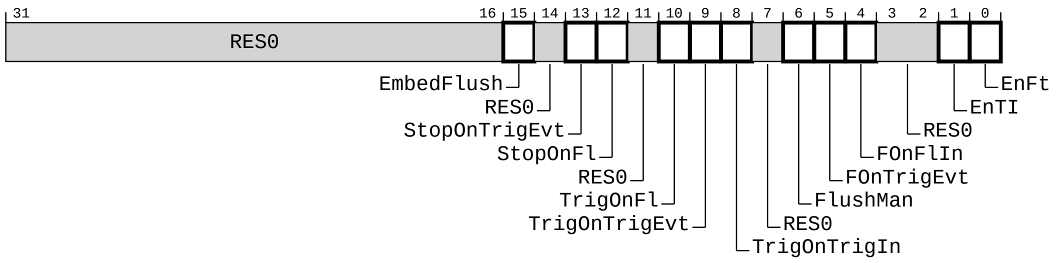
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-259: css600_tmc_etb_FFCR



The following table shows the bit assignments.

Table 10-269: FFCR attributes

Bits	Reset value	Name	Type	Function
[31:16]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[15]	`0b0`	EmbedFlush	Read-write	Embed Flush ID (flush completion packet). Enables insertion of Flush ID `0x7B` with a single byte of data payload = `0x00` in the output trace, immediately after the last flush data byte, when a flush completes on the ATB slave interface. This bit is effective only in Normal formatting modes. In Bypass mode, the Flush ID insertion remains disabled and this bit is ignored. **0** Disable Flush ID insertion. **1** Enable Flush ID insertion.
[14]	`0b0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[13]	`0b0`	StopOnTrigEvt	Read-write	Stop On Trigger Event. If this bit is set, the formatter is stopped when a Trigger Event has been observed. This bit must be used only in CB mode because in FIFO modes, the TMC is a trace link rather than a trace sink and trigger events are related to trace sink functionality. If trace capture is enabled in SWF1 mode with this bit set, it results in **UNPREDICTABLE** behavior.
[12]	`0b0`	StopOnFl	Read-write	Stop On Flush. If this bit is set, the formatter is stopped on completion of a flush operation. The initiation of a flush operation is controlled by programming the register bits FFCR.FlushMan, FFCR.FOnTrigEvt, and FFCR.FOnFlIn. When a flush-initiation condition occurs, afvalid_s is asserted, and when the flush completion is received, that is, afready_s=1, trace capture is stopped. Any remaining data in the formatter is appended with a post-amble and written to trace memory. The flush operation is then complete.
[11]	`0b0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[10]	`0b0`	TrigOnFl	Read-write	Indicate on trace stream the completion of flush. If this bit is set, a trigger is indicated on the trace stream when afready_s is received for a flush in progress. If this bit is clear, no triggers are embedded in the trace stream on flush completion. If Trigger Insertion is disabled, that is, FFCR.EnTI=0, then trigger indication on the trace stream is blocked regardless of the value that is programmed in this bit.
[9]	`0b0`	TrigOnTrigEvt	Read-write	Indicate on trace stream the occurrence of a Trigger Event. If this bit is set, a trigger is indicated on the output trace stream when a Trigger Event occurs. If Trigger Insertion is disabled, that is, FFCR.EnTI=0, then trigger indication on the trace stream is blocked regardless of the value that is programmed in this bit. This bit must be used only in CB mode because in FIFO modes, the TMC is a trace link rather than a trace sink and trigger events are related to trace sink functionality. If trace capture is enabled in SWF1 mode with this bit set, it results in **UNPREDICTABLE** behavior.
[8]	`0b0`	TrigOnTrigIn	Read-write	Indicate on trace stream the occurrence of a rising edge on trigin. If this bit is set, a trigger is indicated on the trace stream when a rising edge is detected on the trigin input. If Trigger Insertion is disabled, that is, FFCR.EnTI=0, then trigger indication on the trace stream is blocked regardless of the value that is programmed in this bit.
[7]	`0b0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[6]	`0b0`	FlushMan	Read-write	Manually generate a flush of the system. Writing 1 to this bit causes a flush to be generated. This bit is cleared automatically when, in formatter bypass mode, afready_s was sampled high, or, in normal formatting mode, afready_s was sampled high and all flush data was output to the trace memory. If CTL.TraceCaptEn=0, writes to this bit are ignored.
[5]	`0b0`	FOnTrigEvt	Read-write	Flush on Trigger Event. If FFCR.StopOnTrigEvt is set, this bit is ignored. Setting this bit generates a flush when a Trigger Event occurs. If FFCR.StopOnTrigEvt is set, this bit is ignored. This bit must be used only in CB mode because in FIFO modes, the TMC is a trace link rather than a trace sink and trigger events are related to trace sink functionality. If trace capture is enabled in SWF1 mode with this bit set, it results in **UNPREDICTABLE** behavior.

Bits	Reset value	Name	Type	Function
[4]	`0b0`	FOnFlIn	Read-write	Setting this bit enables the detection of transitions on the flushin input by the TMC. If this bit is set and the formatter has not already stopped, a rising edge on flushin initiates a flush request.
[3:2]	`0b00`	RES0	Read-write	Reserved bit or field with SBZP behavior
[1]	`0b0`	EnTI	Read-write	Enable Trigger Insertion. Setting this bit enables the insertion of triggers in the formatted trace stream. A trigger is indicated by inserting one byte of data `0x00` with atid_s=`0x7D` in the trace stream. Trigger indication on the trace stream is also controlled by the register bits FFCR.TrigOnFl, FFCR.TrigOnTrigEvt, and FFCR.TrigOnTrigIn. This bit can only be changed when the TMC is in Disabled state. If FFCR.EnTI bit is set formatting is enabled.
[0]	`0b0`	EnFt	Read-write	Enable Formatter. If this bit is set, formatting is enabled. When EnTi is set, formatting is enabled. When CB mode is not used, formatting is also enabled. For backwards-compatibility with earlier versions of the ETB disabling of formatting is supported only in CB mode. This bit can only be changed when TMC is in Disabled state.

10.11.2.15 css600_tmc_etb Periodic Synchronization Counter Register, PSCR

This register determines the reload value of the Periodic Synchronization Counter. This counter enables the frequency of sync packets to be optimized to the trace capture buffer size. The default behavior of the counter is to generate periodic synchronization requests, syncreq_s, on the ATB slave interface.

Attributes

Offset

`0x0308`

Type

Read-write

Reset

`0x0000000A`

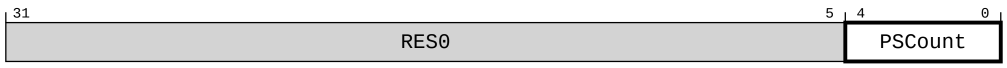
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-260: css600_tmc_etb_PSCR



The following table shows the bit assignments.

Table 10-270: PSCR attributes

Bits	Reset value	Name	Type	Function
[31:5]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[4:0]	`0b01010`	PSCount	Read-write	Periodic Synchronization Count. Determines the reload value of the Synchronization Counter. The reload value takes effect the next time the counter reaches zero. When trace capture is enabled, the Synchronization Counter counts the number of bytes of trace data that is stored into the trace memory, regardless of whether the trace data has been formatted by the TMC or not, since the occurrence of the last sync request on the ATB slave interface. When the counter reaches 0, a sync request is sent on the ATB slave interface. Reads from this register return the reload value that is programmed in this register. This field resets to `0x0A`, that is, the default sync period is 2^10 bytes. If a reserved value is programmed in this register field, the value `0x1B` is used instead, and subsequent reads from this register also return `0x1B`. The following constraints apply to the values written to the PSCount field: `0x0` - synchronization is disabled, `0x1`-`0x6` - reserved, `0x7`-`0x1B` - synchronization period is 2^PSCount bytes. The smallest value `0x7` gives a sync period of 128 bytes. The maximum allowed value `0x1B` gives a sync period of 2^27 bytes, `0x1C`-`0x1F` - reserved.

10.11.2.16 css600_tmc_etb Integration Test Event and Interrupt Control Register, ITEVTINTR

This register controls the values of event and interrupt outputs in integration mode. In functional mode, this register behaves as RAZ/WI. In integration mode, the value that is written to any bit of this register is driven on the output pin that is controlled by that bit and the reads return `0x0`.

Attributes

Offset

`0x0EE0`

Type

Write-only

Reset

`0x00000000`

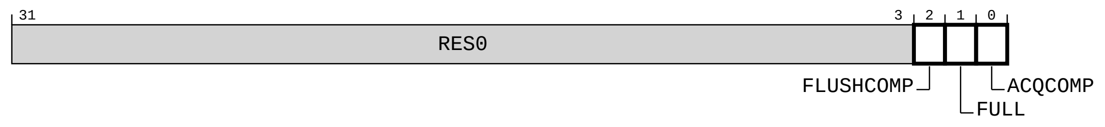
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-261: css600_tmc_etb_ITEVTINTR



The following table shows the bit assignments.

Table 10-271: ITEVTINTR attributes

Bits	Reset value	Name	Type	Function
[31:3]	`0x0`	RES0	Write-only	Reserved bit or field with SBZP behavior
[2]	`0b0`	FLUSHCOMP	Write-only	Controls the value of flushcomp output in integration mode.
[1]	`0b0`	FULL	Write-only	Controls the value of full output in integration mode.
[0]	`0b0`	ACQCOMP	Write-only	Controls the value of acqcomp output in integration mode.

10.11.2.17 css600_tmc_etb Integration Test Trigger In and Flush In register, ITTRFLIN

This register captures the values of the flushin and trigin inputs in integration mode. In functional mode, this register behaves as RAZ/WI.

Attributes

Offset

`0x0EE8`

Type

Read-only

Reset

`0x00000000`

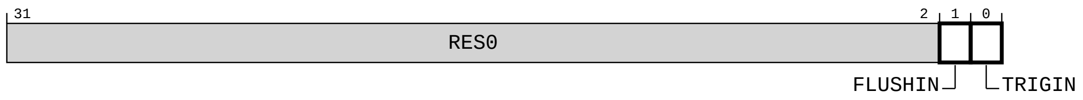
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-262: css600_tmc_etb_ITTRFLIN



The following table shows the bit assignments.

Table 10-272: ITTRFLIN attributes

Bits	Reset value	Name	Type	Function
[31:2]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[1]	`0b0`	FLUSHIN	Read-only	Integration status of flushin input. In integration mode, this bit latches to 1 on a rising edge of the flushin input. It is cleared when the register is read or when integration mode is disabled.
[0]	`0b0`	TRIGIN	Read-only	Integration status of trigin input. In integration mode, this bit latches to 1 on a rising edge of the trigin input. It is cleared when the register is read or when integration mode is disabled.

10.11.2.18 css600_tmc_etb Integration Test ATB Data 0 Register, ITATBDATA0

This register captures the value of atdata_s input in integration mode. In functional mode, this register behaves as RAZ/WI. In integration mode, writes to this register are ignored and the reads return the value of corresponding atdata_s bits. The width of this register is given by: 1+(ATB DATA WIDTH)/8.

Attributes

Offset

`0x0EEC`

Type

Read-only

Reset

`0x00000000`

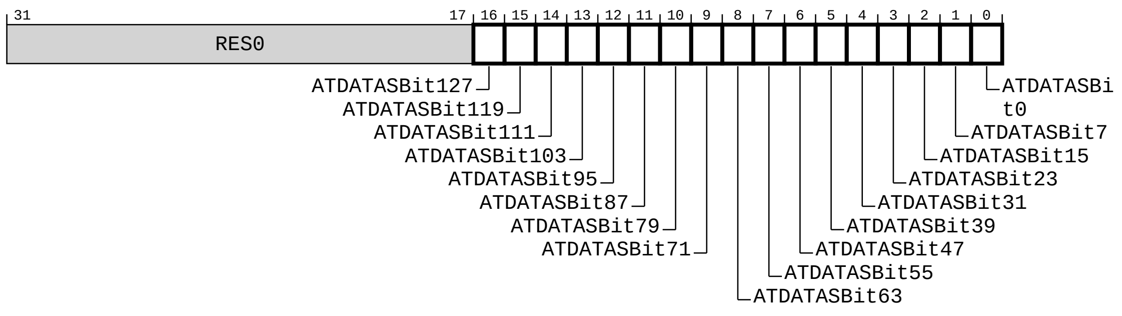
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-263: css600_tmc_etb_ITATBDATA0



The following table shows the bit assignments.

Table 10-273: ITATBDATA0 attributes

Bits	Reset value	Name	Type	Function
[31:17]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[16]	`0b0`	ATDATASBit127	Read-only	Returns the value of atdata_s[127] input in integration mode.
[15]	`0b0`	ATDATASBit119	Read-only	Returns the value of atdata_s[119] input in integration mode.
[14]	`0b0`	ATDATASBit111	Read-only	Returns the value of atdata_s[111] input in integration mode.
[13]	`0b0`	ATDATASBit103	Read-only	Returns the value of atdata_s[103] input in integration mode.
[12]	`0b0`	ATDATASBit95	Read-only	Returns the value of atdata_s[95] input in integration mode.
[11]	`0b0`	ATDATASBit87	Read-only	Returns the value of atdata_s[87] input in integration mode.

Bits	Reset value	Name	Type	Function
[10]	`0b0`	ATDATASBit79	Read-only	Returns the value of atdata_s[79] input in integration mode.
[9]	`0b0`	ATDATASBit71	Read-only	Returns the value of atdata_s[71] input in integration mode.
[8]	`0b0`	ATDATASBit63	Read-only	Returns the value of atdata_s[63] input in integration mode.
[7]	`0b0`	ATDATASBit55	Read-only	Returns the value of atdata_s[55] input in integration mode.
[6]	`0b0`	ATDATASBit47	Read-only	Returns the value of atdata_s[47] input in integration mode.
[5]	`0b0`	ATDATASBit39	Read-only	Returns the value of atdata_s[39] input in integration mode.
[4]	`0b0`	ATDATASBit31	Read-only	Returns the value of atdata_s[31] input in integration mode.
[3]	`0b0`	ATDATASBit23	Read-only	Returns the value of atdata_s[23] input in integration mode.
[2]	`0b0`	ATDATASBit15	Read-only	Returns the value of atdata_s[15] input in integration mode.
[1]	`0b0`	ATDATASBit7	Read-only	Returns the value of atdata_s[7] input in integration mode.
[0]	`0b0`	ATDATASBit0	Read-only	Returns the value of atdata_s[0] input in integration mode.

10.11.2.19 css600_tmc_etb Integration Test ATB Control 2 Register, ITATBCTR2

This register enables control of ATB slave outputs atready_s, afvalid_s, and syncreq_s in integration mode. In functional mode, this register behaves as RAZ/WI. In integration mode, the value that is written to any bit of this register is driven on the output pin that is controlled by that bit and the reads return `0x0`.

Attributes

Offset

`0x0EF0`

Type

Write-only

Reset

`0x00000000`

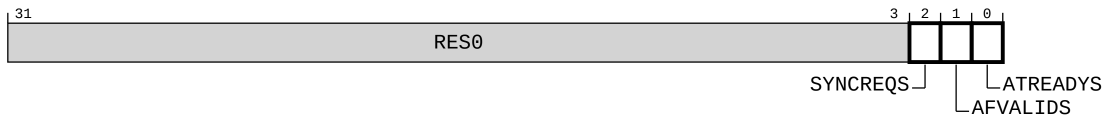
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-264: css600_tmc_etb_ITATBCTR2



The following table shows the bit assignments.

Table 10-274: ITATBCTR2 attributes

Bits	Reset value	Name	Type	Function
[31:3]	`0x0`	RES0	Write-only	Reserved bit or field with SBZP behavior
[2]	`0b0`	SYNCREQS	Write-only	Controls the value of syncreq_s output in integration mode.
[1]	`0b0`	AFVALIDS	Write-only	Controls the value of afvalid_s output in integration mode.
[0]	`0b0`	ATREADYS	Write-only	Controls the value of atready_s output in integration mode.

10.11.2.20 css600_tmc_etb Integration Test ATB Control 1 Register, ITATBCTR1

This register captures the value of the atid_s[6:0] input in integration mode. In functional mode, this register behaves as RAZ/WI. In integration mode, writes to this register are ignored and the reads return the value of atid_s input.

Attributes

Offset

`0x0EF4`

Type

Read-only

Reset

`0x00000000`

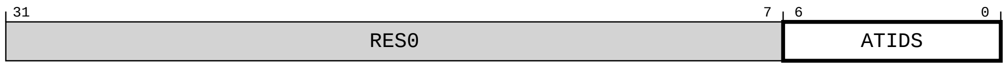
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-265: css600_tmc_etb_ITATBCTR1



The following table shows the bit assignments.

Table 10-275: ITATBCTR1 attributes

Bits	Reset value	Name	Type	Function
[31:7]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[6:0]	`0x0`	ATIDS	Read-only	Returns the value of atid_s[6:0] input in integration mode.

10.11.2.21 css600_tmc_etb Integration Test ATB Control 0 Register, ITATBCTR0

This register captures the values of ATB slave inputs atvalid_s, afready_s, atwakeup_s, and atbytes_s in integration mode. In functional mode, this register behaves as RAZ/WI. In integration mode, writes to this register are ignored and the reads return the value of corresponding input pins. The width of this register is given by: 8+log2(ATB DATA WIDTH/8).

Attributes

Offset

`0x0EF8`

Type

Read-only

Reset

`0x00000000`

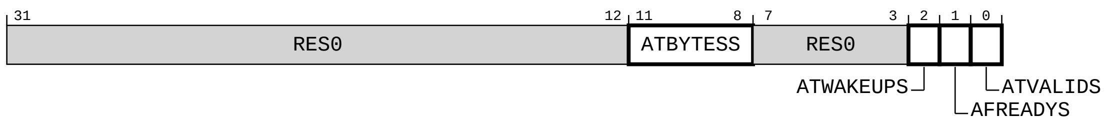
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-266: css600_tmc_etb_ITATBCTR0



The following table shows the bit assignments.

Table 10-276: ITATBCTR0 attributes

Bits	Reset value	Name	Type	Function
[31:12]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[11:8]	`0b0000`	ATBYTESS	Read-only	Returns the value of atbytes_s input in integration mode. N=8+log2(ATB DATA WIDTH/8).
[7:3]	`0b00000`	RES0	Read-only	Reserved bit or field with SBZP behavior
[2]	`0b0`	ATWAKEUPS	Read-only	Returns the value of atwakeup_s input in integration mode.
[1]	`0b0`	AFREADYS	Read-only	Returns the value of afready_s input in integration mode.
[0]	`0b0`	ATVALIDS	Read-only	Returns the value of atvalid_s input in integration mode.

10.11.2.22 css600_tmc_etb Integration Mode Control Register, ITCTRL

The Integration Mode Control register is used to enable topology detection.

Attributes

Offset

`0x0F00`

Type

Read-write

Reset

`0x00000000`

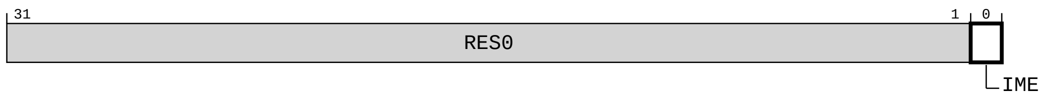
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-267: css600_tmc_etb_ITCTRL



The following table shows the bit assignments.

Table 10-277: ITCTRL attributes

Bits	Reset value	Name	Type	Function
[31:1]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[0]	`0b0`	IME	Read-write	Integration Mode Enable. When set, the component enters integration mode, enabling topology detection or integration testing to be performed.

10.11.2.23 css600_tmc_etb Claim Tag Set Register, CLAIMSET

This register forms one half of the claim tag value. On writes, this location enables individual bits to be set. On reads, it returns the number of bits that can be set.

Attributes

Offset

`0x0FA0`

Type

Read-write

Reset

`0x0000000F`

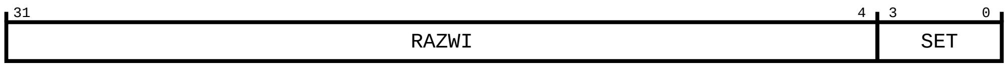
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-268: css600_tmc_etb_CLAIMSET



The following table shows the bit assignments.

Table 10-278: CLAIMSET attributes

Bits	Reset value	Name	Type	Function
[31:4]	`0x0`	RAZ/WI	Read-write	RAZ/WI
[3:0]	`0b1111`	SET	Read-write	A bit-programmable register bank that sets the claim tag value. A read returns a logic 1 for all implemented locations.

10.11.2.24 css600_tmc_etb Claim Tag Clear Register, CLAIMCLR

This register forms one half of the claim tag value. On writes, this location enables individual bits to be cleared. On reads, it returns the current claim tag value.

Attributes

Offset

`0x0FA4`

Type

Read-write

Reset

`0x00000000`

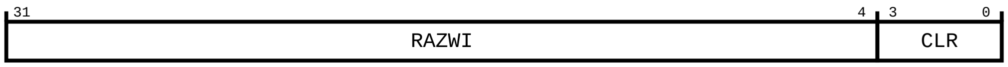
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-269: css600_tmc_etb_CLAIMCLR



The following table shows the bit assignments.

Table 10-279: CLAIMCLR attributes

Bits	Reset value	Name	Type	Function
[31:4]	`0x0`	RAZ/WI	Read-write	RAZ/WI
[3:0]	`0b0000`	CLR	Read-write	A bit-programmable register bank that clears the claim tag value. It is zero at reset. It is used by software agents to signal to each other ownership of the hardware. It has no direct effect on the hardware itself.

10.11.2.25 css600_tmc_etb Authentication Status Register, AUTHSTATUS

Reports the current status of the authentication control signals.

Attributes

Offset

`0x0FB8`

Type

Read-only

Reset

`0x00000000`

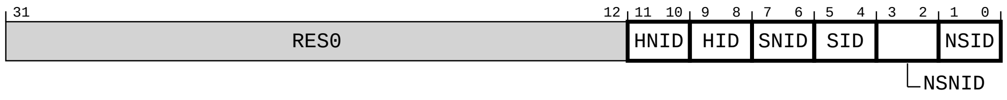
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-270: css600_tmc_etb_AUTHSTATUS



The following table shows the bit assignments.

Table 10-280: AUTHSTATUS attributes

Bits	Reset value	Name	Type	Function
[31:12]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[11:10]	`0b00`	HNID	Read-only	Hypervisor non-invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.
[9:8]	`0b00`	HID	Read-only	Hypervisor invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.
[7:6]	`0b00`	SNID	Read-only	Secure non-invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.
[5:4]	`0b00`	SID	Read-only	Secure invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.
[3:2]	`0b00`	NSNID	Read-only	Non-secure non-invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.
[1:0]	`0b00`	NSID	Read-only	Non-secure invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.

10.11.2.26 css600_tmc_etb Device Configuration Register 1, DEVID1

Contains an **IMPLEMENTATION DEFINED** value.

Attributes

Offset

`0x0FC4`

Type

Read-only

Reset

`0x00000001`

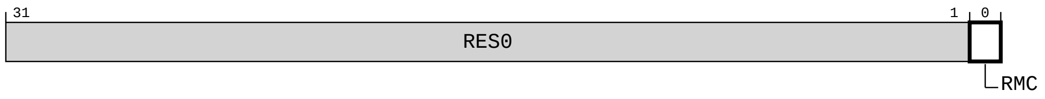
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-271: css600_tmc_etb_DEVID1



The following table shows the bit assignments.

Table 10-281: DEVID1 attributes

Bits	Reset value	Name	Type	Function
[31:1]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[0]	`0b1`	RMC	Read-only	Register management mode. TMC implements register management mode 1.

10.11.2.27 css600_tmc_etb Device Configuration Register, DEVID

This register is **IMPLEMENTATION DEFINED** for each Part Number and Designer. The register indicates the capabilities of the component.

Attributes

Offset

`0x0FC8`

Type

Read-only

Reset

`0x00000-00`

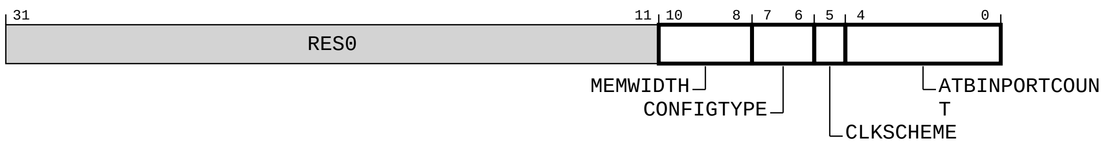
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-272: css600_tmc_etb_DEVID



The following table shows the bit assignments.

Table 10-282: DEVID attributes

Bits	Reset value	Name	Type	Function
[31:11]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[10:8]	IMPLEMENTATION DEFINED	MEMWIDTH	Read-only	This value is twice ATB_DATA_WIDTH. **`0x3`** Memory interface databus is 64 bits wide. (ATB_DATA_WIDTH = 32bit) **`0x4`** Memory interface databus is 128 bits wide. (ATB_DATA_WIDTH = 64bit) **`0x5`** Memory interface databus is 256 bits wide. (ATB_DATA_WIDTH = 128bit)
[7:6]	`0b00`	CONFIGTYPE	Read-only	Returns `0x0`, indicating ETB configuration.
[5]	`0b0`	CLKSCHEME	Read-only	RAM Clocking Scheme. This value indicates the TMC RAM clocking scheme used, that is, whether the TMC RAM operates synchronously or asynchronously to the TMC clock. Fixed to 0 indicating that TMC RAM clock is synchronous to the clk input.
[4:0]	`0b00000`	ATBINPORTCOUNT	Read-only	Hidden Level of ATB input multiplexing. This value indicates the type/number of ATB multiplexing present on the input ATB. Fixed to `0x00` indicating that no multiplexing is present.

10.11.2.28 css600_tmc_etb Device Type Identifier Register, DEVTYPE

A debugger can use this register to get information about a component that has an unrecognized Part number.

Attributes

Offset

`0x0FCC`

Type

Read-only

Reset

`0x00000021`

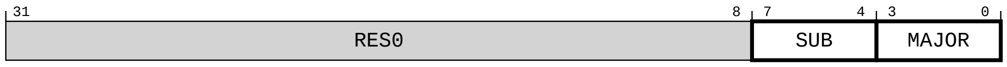
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-273: css600_tmc_etb_DEVTYPE



The following table shows the bit assignments.

Table 10-283: DEVTYPE attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0010`	SUB	Read-only	Minor classification. Returns `0x2`, indicating this component is a Buffer.
[3:0]	`0b0001`	MAJOR	Read-only	Major classification. Returns `0x1`, indicating this component is a Trace Sink.

10.11.2.29 css600_tmc_etb Peripheral Identification Register 4, PIDR4

The PIDR4 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FD0`

Type

Read-only

Reset

`0x00000004`

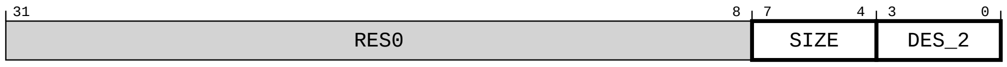
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-274: css600_tmc_etb_PIDR4



The following table shows the bit assignments.

Table 10-284: PIDR4 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0000`	SIZE	Read-only	Indicates the memory size that is used by this component. Returns 0 indicating that the component uses an **UNKNOWN** number of 4KB blocks. Using the SIZE field to indicate the size of the component is deprecated.
[3:0]	`0b0100`	DES_2	Read-only	JEP106 continuation code. Together, with PIDR2.DES_1 and PIDR1.DES_0, they indicate the designer of the component and not the implementer, except where the two are the same.

10.11.2.30 css600_tmc_etb Peripheral Identification Register 5, PIDR5

The PIDR5 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FD4`

Type

Read-only

Reset

`0x00000000`

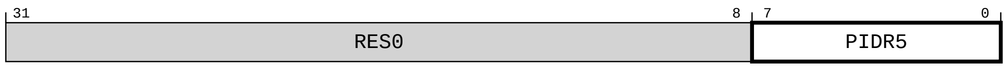
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-275: css600_tmc_etb_PIDR5



The following table shows the bit assignments.

Table 10-285: PIDR5 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	PIDR5	Read-only	Reserved.

10.11.2.31 css600_tmc_etb Peripheral Identification Register 6, PIDR6

The PIDR6 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FD8`

Type

Read-only

Reset

`0x00000000`

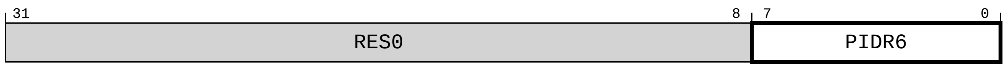
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-276: css600_tmc_etb_PIDR6



The following table shows the bit assignments.

Table 10-286: PIDR6 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	PIDR6	Read-only	Reserved.

10.11.2.32 css600_tmc_etb Peripheral Identification Register 7, PIDR7

The PIDR7 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FDC`

Type

Read-only

Reset

`0x00000000`

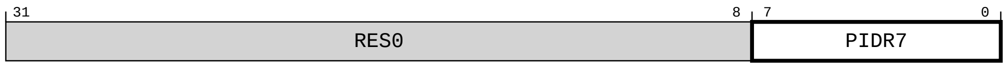
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-277: css600_tmc_etb_PIDR7



The following table shows the bit assignments.

Table 10-287: PIDR7 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	PIDR7	Read-only	Reserved.

10.11.2.33 css600_tmc_etb Peripheral Identification Register 0, PIDR0

The PIDR0 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FE0`

Type

Read-only

Reset

`0x000000E9`

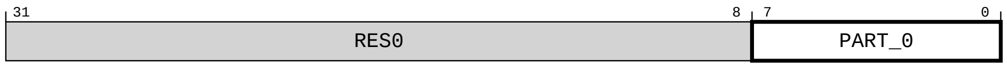
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-278: css600_tmc_etb_PIDR0



The following table shows the bit assignments.

Table 10-288: PIDR0 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0xE9`	PART_0	Read-only	Part number (lower 8 bits). Returns 0xe9, indicating TMC ETB.

10.11.2.34 css600_tmc_etb Peripheral Identification Register 1, PIDR1

The PIDR1 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FE4`

Type

Read-only

Reset

`0x000000B9`

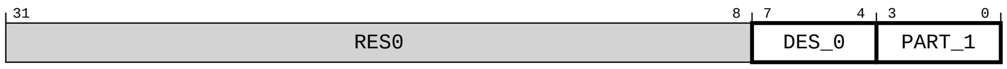
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-279: css600_tmc_etb_PIDR1



The following table shows the bit assignments.

Table 10-289: PIDR1 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b1011`	DES_0	Read-only	JEP106 identification code, bits[3:0]. Together, with PIDR4.DES_2 and PIDR2.DES_1, they indicate the designer of the component and not the implementer, except where the two are the same.
[3:0]	`0b1001`	PART_1	Read-only	Part number, bits[11:8]. Taken together with PIDR0.PART_0 it indicates the component. The Part Number is selected by the designer of the component.

10.11.2.35 css600_tmc_etb Peripheral Identification Register 2, PIDR2

The PIDR2 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FE8`

Type

Read-only

Reset

`0x0000006B`

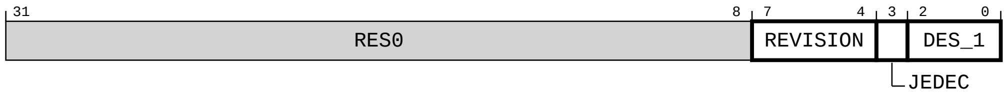
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-280: css600_tmc_etb_PIDR2



The following table shows the bit assignments.

Table 10-290: PIDR2 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0110`	REVISION	Read-only	Revision. It is an incremental value starting at `0x0` for the first design of a component. See the Component list in Chapter 1 for information on the RTL revision of the component.
[3]	`0b1`	JEDEC	Read-only	1 - Always set. Indicates that a JEDEC assigned value is used.
[2:0]	`0b011`	DES_1	Read-only	JEP106 identification code, bits[6:4]. Together, with PIDR4.DES_2 and PIDR1.DES_0, they indicate the designer of the component and not the implementer, except where the two are the same.

10.11.2.36 css600_tmc_etb Peripheral Identification Register 3, PIDR3

The PIDR3 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FEC`

Type

Read-only

Reset

`0x00000000`

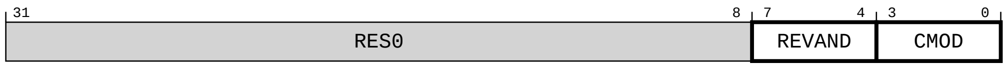
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-281: css600_tmc_etb_PIDR3



The following table shows the bit assignments.

Table 10-291: PIDR3 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0000`	REVAND	Read-only	This field indicates minor errata fixes specific to this design, for example metal fixes after implementation. In most cases this field is `0x0`.
[3:0]	`0b0000`	CMOD	Read-only	Customer Modified. Where the component is reusable IP, this value indicates if the customer has modified the behavior of the component. In most cases this field is `0x0`.

10.11.2.37 css600_tmc_etb Component Identification Register 0, CIDR0

The CIDR0 register is part of the set of component identification registers.

Attributes

Offset

`0x0FF0`

Type

Read-only

Reset

`0x0000000D`

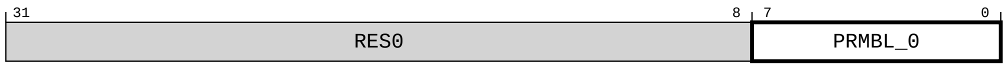
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-282: css600_tmc_etb_CIDR0



The following table shows the bit assignments.

Table 10-292: CIDR0 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0xD`	PRMBL_0	Read-only	Preamble. Returns `0x0D`.

10.11.2.38 css600_tmc_etb Component Identification Register 1, CIDR1

The CIDR1 register is part of the set of component identification registers.

Attributes

Offset

`0x0FF4`

Type

Read-only

Reset

`0x00000090`

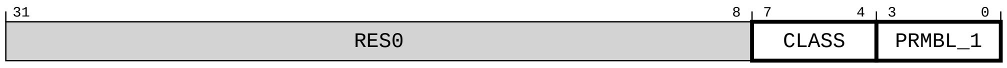
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-283: css600_tmc_etb_CIDR1



The following table shows the bit assignments.

Table 10-293: CIDR1 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b1001`	CLASS	Read-only	Component class. Returns `0x9`, indicating this is a CoreSight component.
[3:0]	`0b0000`	PRMBL_1	Read-only	Preamble. Returns `0x0`.

10.11.2.39 css600_tmc_etb Component Identification Register 2, CIDR2

The CIDR2 register is part of the set of component identification registers.

Attributes

Offset

`0x0FF8`

Type

Read-only

Reset

`0x00000005`

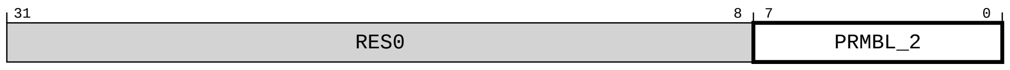
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-284: css600_tmc_etb_CIDR2



The following table shows the bit assignments.

Table 10-294: CIDR2 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x5`	PRMBL_2	Read-only	Preamble. Returns `0x05`.

10.11.2.40 css600_tmc_etb Component Identification Register 3, CIDR3

The CIDR3 register is part of the set of component identification registers.

Attributes

Offset
`0x0FFC`

Type
Read-only

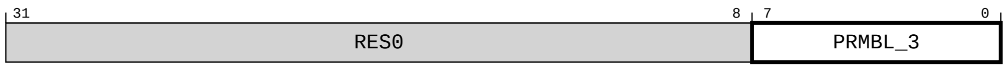
Reset
`0x000000B1`

Width
32

Bit descriptions

The following image shows the bit assignments.

Figure 10-285: css600_tmc_etb_CIDR3



The following table shows the bit assignments.

Table 10-295: CIDR3 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0xB1`	PRMBL_3	Read-only	Preamble. Returns `0xB1`.

10.12 `css600_tpiu` introduction

This section describes the programmers model of the `css600_tpiu`.

10.12.1 css600_tpiu register summary

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

Note

A reset value containing one or more '-' means that this register contains **UNKNOWN** or **IMPLEMENTATION-DEFINED** values. See the relevant register description for more information.

Locations that are not listed in the table are Reserved.

Table 10-296: css600_tpiu register summary

Offset	Name	Type	Reset	Width	Description
0x0000	SSPSR	RO	0x--------	32	css600_tpiu Supported Port Size Register, SSPSR
0x0004	CSPSR	RW	0x00000001	32	css600_tpiu Current Port Size Register, CSPSR
0x0100	STMR	RO	0x0000011F	32	css600_tpiu Supported Trigger Modes Register, STMR
0x0104	TCVR	RW	0x00000000	32	css600_tpiu Trigger Counter Value Register, TCVR
0x0108	TCMR	RW	0x00000000	32	css600_tpiu Trigger Counter Multiplier Register, TCMR
0x0200	STPMR	RO	0x0003000F	32	css600_tpiu Supported Test Patterns/Modes Register, STPMR
0x0204	CTPMR	RW	0x00000000	32	css600_tpiu Current Test Patterns/Modes Register, CTPMR
0x0208	TPRCR	RW	0x00000000	32	css600_tpiu Test Pattern Repeat Counter Register, TPRCR
0x0300	FFSR	RO	0x0000000-	32	css600_tpiu Formatter and Flush Status Register, FFSR
0x0304	FFCR	RW	0x00001000	32	css600_tpiu Formatter and Flush Control Register, FFCR
0x0308	FSCR	RW	0x00000040	32	css600_tpiu Formatter Synchronization Count Register, FSCR
0x0400	EXTCTLIN	RO	0x000000--	32	css600_tpiu External Control Port In Register, EXTCTLIN
0x0404	EXTCTLOUT	RW	0x00000000	32	css600_tpiu External Control Port Out Register, EXTCTLOUT
0x0EE8	ITTRFLIN	RO	0x00000000	32	css600_tpiu Integration Test Trigger In and Flush In Register, ITTRFLIN
0x0EEC	ITATBDATA0	RO	0x00000000	32	css600_tpiu Integration Test ATB Data Register 0, ITATBDATA0
0x0EF0	ITATBCTR2	WO	0x00000000	32	css600_tpiu Integration Test ATB Control Register 2, ITATBCTR2
0x0EF4	ITATBCTR1	RO	0x00000000	32	css600_tpiu Integration Test ATB Control Register 1, ITATBCTR1
0x0EF8	ITATBCTR0	RO	0x00000000	32	css600_tpiu Integration Test ATB Control Register 0, ITATBCTR0
0x0EFC	ITOUTCTR	WO	0x00000000	32	css600_tpiu Integration Test Output Control Register, ITOUTCTR
0x0F00	ITCTRL	RW	0x00000000	32	css600_tpiu Integration Mode Control Register, ITCTRL
0x0FA0	CLAIMSET	RW	0x0000000F	32	css600_tpiu Claim Tag Set Register, CLAIMSET
0x0FA4	CLAIMCLR	RW	0x00000000	32	css600_tpiu Claim Tag Clear Register, CLAIMCLR
0x0FB8	AUTHSTATUS	RO	0x00000000	32	css600_tpiu Authentication Status Register, AUTHSTATUS
0x0FBC	DEVARCH	RO	0x00000000	32	css600_tpiu Device Architecture Register, DEVARCH
0x0FC8	DEVID	RO	0x00000020	32	css600_tpiu Device Configuration Register, DEVID
0x0FCC	DEVTYPE	RO	0x00000011	32	css600_tpiu Device Type Identifier Register, DEVTYPE
0x0FD0	PIDR4	RO	0x00000004	32	css600_tpiu Peripheral Identification Register 4, PIDR4
0x0FD4	PIDR5	RO	0x00000000	32	css600_tpiu Peripheral Identification Register 5, PIDR5
0x0FD8	PIDR6	RO	0x00000000	32	css600_tpiu Peripheral Identification Register 6, PIDR6
0x0FDC	PIDR7	RO	0x00000000	32	css600_tpiu Peripheral Identification Register 7, PIDR7
0x0FE0	PIDR0	RO	0x000000E7	32	css600_tpiu Peripheral Identification Register 0, PIDR0
0x0FE4	PIDR1	RO	0x000000B9	32	css600_tpiu Peripheral Identification Register 1, PIDR1
0x0FE8	PIDR2	RO	0x0000002B	32	css600_tpiu Peripheral Identification Register 2, PIDR2
0x0FEC	PIDR3	RO	0x00000000	32	css600_tpiu Peripheral Identification Register 3, PIDR3
0x0FF0	CIDR0	RO	0x0000000D	32	css600_tpiu Component Identification Register 0, CIDR0
0x0FF4	CIDR1	RO	0x00000090	32	css600_tpiu Component Identification Register 1, CIDR1
0x0FF8	CIDR2	RO	0x00000005	32	css600_tpiu Component Identification Register 2, CIDR2

Offset	Name	Type	Reset	Width	Description
`0x0FFC`	CIDR3	RO	`0x000000B1`	32	css600_tpiu Component Identification Register 3, CIDR3

10.12.2 Register descriptions

This section describes the `css600_tpiu` registers.

css600_tpiu register summary provides cross references to individual registers.

10.12.2.1 css600_tpiu Supported Port Size Register, SSPSR

The SSPSR register is **IMPLEMENTATION DEFINED**. It shows supported width configurations of the tracedata port. Each bit location represents a single port size that is supported, that is sizes 32 bits down to 1 bit, in bit locations [31:0]. If a bit is set, then that port size is supported. By default, the RTL is designed to support all port sizes. Port sizes, other than 1-bit, are configuration-dependent on the tie-off value of tp_maxdatasize. Bit[0] is always 1.

Attributes

Offset

`0x0000`

Type

Read-only

Reset

`0x--------`

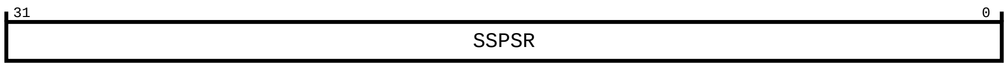
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-286: css600_tpiu_SSPSR



The following table shows the bit assignments.

Table 10-297: SSPSR attributes

Bits	Reset value	Name	Type	Function
[31:0]	IMPLEMENTATION DEFINED	SSPSR	Read-only	Supported tracedata port sizes. Bit[0] is always 1.

10.12.2.2 css600_tpiu Current Port Size Register, CSPSR

The CSPSR register shows the currently selected size of the tracedata port. It has the same format as the Supported Port Size Register but only one bit is set to show the currently selected port size. If a bit that is indicated as not supported in the SSPSR is set in the CSPSR, it can corrupt the output trace stream, in trace capture mode, and the trace patterns in pattern generation mode. If more than one bit is set, this register indicates the programmed size. However, the port size is internally resolved to the highest order set bit. This register must not be modified while the trace port is still active, or without correctly stopping the formatter. If this happens, it can result in data not being aligned to the port width, for example, data on an 8-bit trace port might not be byte aligned. For the register access to complete on APB clocking on traceclk_in is needed.

Attributes

Offset

`0x0004`

Type

Read-write

Reset

`0x00000001`

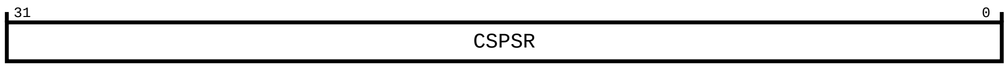
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-287: css600_tpiu_CSPSR



The following table shows the bit assignments.

Table 10-298: CSPSR attributes

Bits	Reset value	Name	Type	Function
[31:0]	`0x1`	CSPSR	Read-write	Currently selected size of the tracedata port

10.12.2.3 css600_tpiu Supported Trigger Modes Register, STMR

The STMR register indicates the implemented Trigger Counter multipliers and other supported features of the trigger system.

Attributes

Offset

`0x0100`

Type

Read-only

Reset

`0x0000011F`

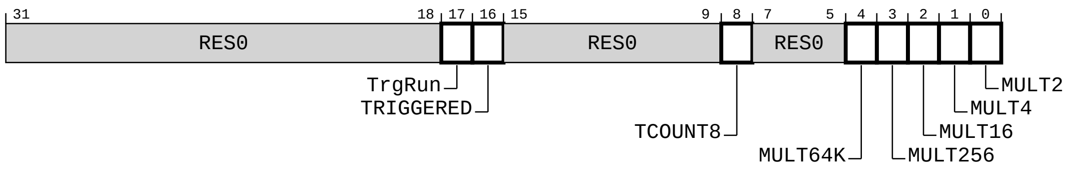
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-288: css600_tpiu_STMR



The following table shows the bit assignments.

Table 10-299: STMR attributes

Bits	Reset value	Name	Type	Function
[31:18]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[17]	`0b0`	TrgRun	Read-only	A trigger has occurred after a rising edge of trigin input, or a trigger packet (atid_s = `0x7D`) is received in the input trace: **0** Either a trigger has not occurred or the counter is at 0 **1** A trigger has occurred but the counter is not at 0
[16]	`0b0`	TRIGGERED	Read-only	A trigger has occurred after a rising edge of trigin input, or a trigger packet (atid_s = `0x7D`) is received in the input trace and the counter has reached 0. This bit is cleared when the TCVR or TCMR register is written. **0** Trigger has not occurred **1** Trigger has occurred
[15:9]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior

Bits	Reset value	Name	Type	Function
[8]	`0b1`	TCOUNT8	Read-only	Returns 1 indicating that an 8-bit wide counter register is implemented for trigger insertion.
[7:5]	`0b000`	RES0	Read-only	Reserved bit or field with SBZP behavior
[4]	`0b1`	MULT64K	Read-only	Returns 1, indicating that multiplying the trigger counter by 65536 is supported
[3]	`0b1`	MULT256	Read-only	Returns 1, indicating that multiplying the trigger counter by 256 is supported
[2]	`0b1`	MULT16	Read-only	Returns 1, indicating that multiplying the trigger counter by 16 is supported
[1]	`0b1`	MULT4	Read-only	Returns 1, indicating that multiplying the trigger counter by 4 is supported
[0]	`0b1`	MULT2	Read-only	Returns 1, indicating that multiplying the trigger counter by 2 is supported

10.12.2.4 css600_tpiu Trigger Counter Value Register, TCVR

The TCVR register indicates the programmed trigger counter value. The TPIU implements a trigger counter that enables delaying the indication of triggers to any external connected trace capture devices. The counter is 8 bits wide and is intended only to be used with the counter multipliers within the Trigger Multiplier Register. When a trigger occurs (observed trigin=1 or atid_s=0x7D), the TCVR.TrigCount value, with the TCMR, determines the number of words to be output from the formatter before the trigger is indicated on the Trace Port. When the trigger counter reaches zero, the value from the TCVR register is reloaded into the trigger counter. Writing to this register causes the trigger counter (the actual counter) to be reloaded. Reading this register returns the programmed count value and not the current count.

Attributes

Offset

`0x0104`

Type

Read-write

Reset

`0x00000000`

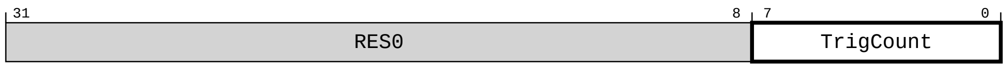
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-289: css600_tpiu_TCVR



The following table shows the bit assignments.

Table 10-300: TCVR attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	TrigCount	Read-write	Programmed trigger counter value.

10.12.2.5 css600_tpiu Trigger Counter Multiplier Register, TCMR

The TCMR register contains the selectors for the trigger counter multiplier. Several multipliers can be selected to create the required multiplier value between 1 (TCMR[4:0] = `0x0`) and 2^31 (TCMR[4:0] = 0x1F). When more than one bit it set, the effective multiplier value is the product of selected multipliers. Writing to this register causes the trigger counter (the actual counter) to be reloaded and the state in the multipliers to be reset.

Attributes

Offset

`0x0108`

Type

Read-write

Reset

`0x00000000`

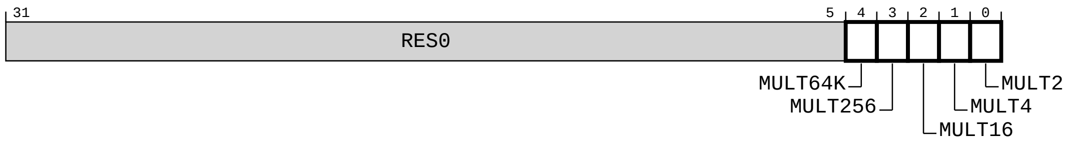
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-290: css600_tpiu_TCMR



The following table shows the bit assignments.

Table 10-301: TCMR attributes

Bits	Reset value	Name	Type	Function
[31:5]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[4]	`0b0`	MULT64K	Read-write	Multiply the Trigger Counter by 65536: **0** Multiplier disabled **1** Multiplier enabled
[3]	`0b0`	MULT256	Read-write	Multiply the Trigger Counter by 256: **0** Multiplier disabled **1** Multiplier enabled
[2]	`0b0`	MULT16	Read-write	Multiply the Trigger Counter by 16: **0** Multiplier disabled **1** Multiplier enabled
[1]	`0b0`	MULT4	Read-write	Multiply the Trigger Counter by 4: **0** Multiplier disabled **1** Multiplier enabled
[0]	`0b0`	MULT2	Read-write	Multiply the Trigger Counter by 2: **0** Multiplier disabled **1** Multiplier enabled

10.12.2.6 css600_tpiu Supported Test Patterns/Modes Register, STPMR

The STPMR Register provides a set of known bit sequences or patterns that can be output over the trace port and can be detected by the TPA or TCD.

Attributes

Offset

`0x0200`

Type

Read-only

Reset

`0x0003000F`

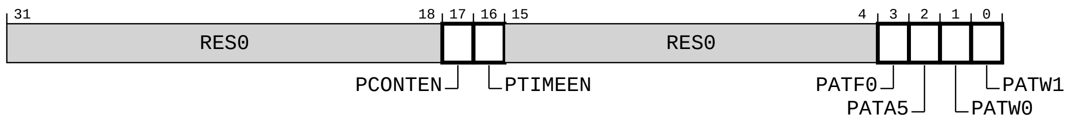
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-291: css600_tpiu_STPMR



The following table shows the bit assignments.

Table 10-302: STPMR attributes

Bits	Reset value	Name	Type	Function
[31:18]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[17]	`0b1`	PCONTEN	Read-only	Continuous Pattern Mode, returns 1 indicating that continuous pattern mode is supported
[16]	`0b1`	PTIMEEN	Read-only	Timed Pattern Mode, returns 1 indicating that timed pattern mode is supported
[15:4]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[3]	`0b1`	PATF0	Read-only	FF/00 Pattern, returns 1 indicating that the FF/00 pattern is supported over the trace port
[2]	`0b1`	PATA5	Read-only	55/AA Pattern, returns 1 indicating that the 55/AA pattern is supported over the trace port
[1]	`0b1`	PATW0	Read-only	Walking 0 Pattern, returns 1 indicating that the walking 0s pattern is supported over the trace port
[0]	`0b1`	PATW1	Read-only	Walking 1s Pattern, returns 1 indicating that the walking 1s pattern is supported over the trace port

10.12.2.7 css600_tpiu Current Test Patterns/Modes Register, CTPMR

The CTPMR register indicates the current test pattern or mode selected. Only one of the two mode bits, bits[17:16], can be set at any one time, but a multiple number of bits for the patterns can be set using bits [3:0]. When timed mode is selected, after the allotted number of cycles is reached, the mode automatically switches to off mode. The pattern with higher bit index is output first when multiple patterns are selected. When no pattern is selected then a default pattern (00/00) is used instead. In continuous mode, the pattern generator continues to send patterns until CTPMR.PCONTEN bit is cleared by software. If multiple patterns are enabled, after sending out all enabled patterns, the pattern generator switches back to the first pattern type and continues to do so until stopped by software. When no pattern is selected then the default pattern (00/00) is used instead. Writing to this register when timed or continuous pattern mode is already enabled causes the current pattern generation to be abandoned and to be restarted with the new pattern mode and new pattern set. Writing to TPRCR or CSPSR when timed or continuous pattern mode is already enabled causes the current pattern generation to be abandoned and to be restarted. The reset value of this register is 0x00000000 which indicates off mode with no selected patterns. For the register access to complete on APB clocking on traceclk_in is needed.

Attributes

Offset

`0x0204`

Type

Read-write

Reset

`0x00000000`

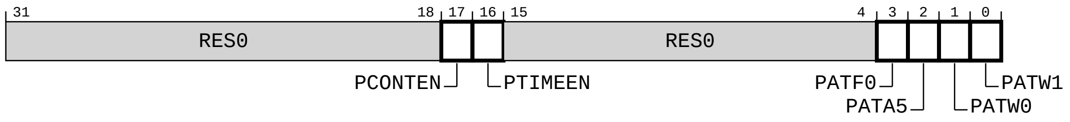
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-292: css600_tpiu_CTPMR



The following table shows the bit assignments.

Table 10-303: CTPMR attributes

Bits	Reset value	Name	Type	Function
[31:18]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[17]	`0b0`	PCONTEN	Read-write	Continuous Pattern Mode. Indicates whether continuous pattern mode is enabled: **0** Mode disabled **1** Mode enabled
[16]	`0b0`	PTIMEEN	Read-write	Timed Pattern Mode. Indicates whether timed pattern mode is enabled: **0** Mode disabled **1** Mode enabled
[15:4]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[3]	`0b0`	PATF0	Read-write	FF/00 Pattern. Indicates whether the FF/00 pattern is enabled as output over the Trace Port. All pins toggle simultaneously as 1-0-1-0. **0** Pattern disabled **1** Pattern enabled

Bits	Reset value	Name	Type	Function
[2]	`0b0`	PATA5	Read-write	55/AA Pattern. Indicates whether the 55/AApattern is enabled as output over the Trace Port. The odd numbered pins toggle as 0-1-0-1, while the even numbered pins toggle as 1-0-1-0, simultaneously. **0** Pattern disabled **1** Pattern enabled
[1]	`0b0`	PATW0	Read-write	Walking 0 Pattern. Indicates whether the walking 0s pattern is enabled as output over the Trace port. To start with, all pins are set to 1, except tracedata[0] which is driven LOW. In each subsequent cycle, the 0 bit shifts to its left by 1 position and eventually rotates around from its starting position, based on the CSPSR value, to tracedata[0], provided the pattern mode remains enabled for a sufficient number of cycles. When timed mode is selected, after the allotted number of cycles is reached (See TPRCR, `0x208`), the wsmode automatically switches to off mode. The pattern with higher bit index is output first when multiple patterns are selected. **0** Pattern disabled **1** Pattern enabled
[0]	`0b0`	PATW1	Read-write	Walking 1s Pattern. Indicates whether the walking 1s pattern is enabled as output over the Trace Port. It is similar to the walking 0s pattern except that tracedata[0] is set to 1 to start with and all other bits are 0. It is this set bit that rotates through all the selected pins of the tracedata port. **0** Pattern disabled **1** Pattern enabled

10.12.2.8 css600_tpiu Test Pattern Repeat Counter Register, TPRCR

This register indicates the number of times each test pattern is output on the Trace Port before switching to next pattern. For the register access to complete on APB clocking on traceclk_in is needed.

Attributes

Offset

`0x0208`

Type

Read-write

Reset

`0x00000000`

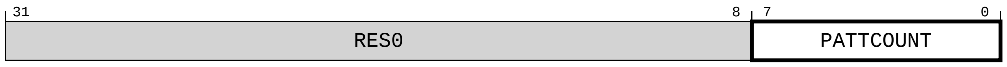
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-293: css600_tpiu_TPRCR



The following table shows the bit assignments.

Table 10-304: TPRCR attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	PATTCOUNT	Read-write	An 8-bit counter value that indicates the number of traceclk cycles for which a pattern runs before it switches to the next enabled pattern. A write sets the initial counter value, and a read returns the programmed value. The pattern length is PATTCOUNT+1. The reset value is `0x0`.

10.12.2.9 css600_tpiu Formatter and Flush Status Register, FFSR

The FFSR indicates the current status of formatter and flush features available in the TPIU.

Attributes

Offset

`0x0300`

Type

Read-only

Reset

`0x0000000-`

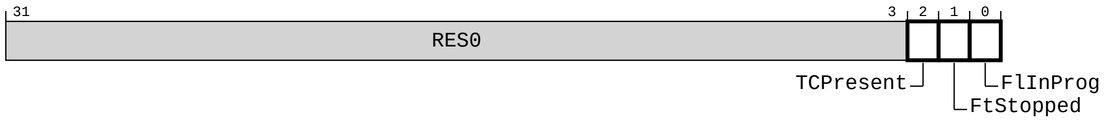
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-294: css600_tpiu_FFSR



The following table shows the bit assignments.

Table 10-305: FFSR attributes

Bits	Reset value	Name	Type	Function
[31:3]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[2]	**IMPLEMENTATION DEFINED**	TCPresent	Read-only	Indicates whether the tracectl pin is available for use, based on the tie-off value of tpctl_valid. **0** tracectl pin not present. The data formatter must be used and in continuous mode. **1** tracectl pin present.
[1]	`0b1`	FtStopped	Read-only	The formatter has received a stop request and all trace data and post-amble is sent. Any further trace on the ATB interface is dropped and atready_s is asserted. **0** Formatter running. **1** Formatter stopped.
[0]	`0b0`	FlInProg	Read-only	Indicates whether a flush is in progress. It is set when the TPIU sends a flush request on its ATB slave interface. The bit remains set until the ATB flush is complete and the last byte of flush data, including the flush ID payload if FFCR.EmbedFlush is set, has been output on the trace port. **0** No ongoing flush. **1** Flush in progress.

10.12.2.10 css600_tpiu Formatter and Flush Control Register, FFCR

The FFCR controls the generation of stop, trigger and flush events. The insertion of a flush completion packet and the insertion of a trigger packet in the formatted trace is enabled here. In bypass mode formatting is disabled and triggers are indicated on tracectl pin, bits[1:0] must be 0b00. In normal mode formatting is enabled and triggers are indicated on tracectl pin, bits[1:0] must be 0b01. In continuous mode formatting is enabled and triggers are embedded in the trace stream with Triger Byte ID 0x7D with a single byte of data payload = 0x00, bits[1:0] must be 0b10. Setting both bits is the same as setting bit[1]. All three flush-generating conditions can be enabled together. However, if a second or third flush event is generated from another condition then the current flush completes before the next flush is serviced. Flush from flushin takes priority over flush from trigger, which in turn completes before a manuallyactivated flush. All trigger indication conditions can be enabled simultaneously although this can cause the appearance of multiple triggers if flush using trigger is also enabled. Both Stop On settings can be enabled although if Flush on Trigger is set up, none of the flushed data is stored. ARM recommends that you change the trace port width without enabling continuous mode. Enabling continuous mode causes data to be sent from the trace port and modifying the port size can result in data not being aligned.

Attributes

Offset

`0x0304`

Type

Read-write

Reset

`0x00001000`

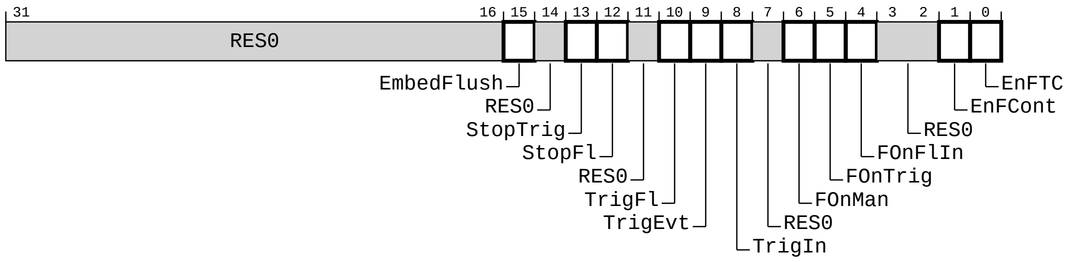
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-295: css600_tpiu_FFCR



The following table shows the bit assignments.

Table 10-306: FFCR attributes

Bits	Reset value	Name	Type	Function
[31:16]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[15]	`0b0`	EmbedFlush	Read-write	Embed flush completion packet, Flush ID. Enables insertion of Flush ID `0x7B` with a single byte of data payload = `0x00` in the output trace, after the last flush data byte, when a flush completes on the ATB slave interface. This bit is effective only in Normal and Continuous formatter modes. In Bypass mode, the Flush ID insertion remains disabled and this bit is ignored. **0** Disable Flush ID insertion **1** Enable Flush ID insertion
[14]	`0b0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[13]	`0b0`	StopTrig	Read-write	Stop on Trigger Event. Stops the formatter after a trigger event is observed. **0** Disable stopping the formatter after a trigger event is observed. **1** Enable stopping the formatter after a trigger event is observed.
[12]	`0b1`	StopFl	Read-write	Stop on Flush Completion. Forces the FIFO to drain off any partially completed packets after a flush completion and stops the formatter. **0** Disable stopping the formatter when afready_s is received. **1** Enable stopping the formatter when afready_s is received.
[11]	`0b0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[10]	`0b0`	TrigFl	Read-write	Trigger on Flush Completion. **0** Disable trigger indication on flush completion, that is, when afready_s is high. **1** Enable trigger indication on flush completion.

Bits	Reset value	Name	Type	Function
[9]	`0b0`	TrigEvt	Read-write	Trigger on Trigger Event. Indicates a trigger when the trigger counter reaches 0 while downcounting. If FFCR.StopTrig is set, this bit is ignored. **0** Disable trigger indication on trigger event. **1** Enable trigger indication on trigger event.
[8]	`0b0`	TrigIn	Read-write	Trigger on trigin. **0** Disable trigger indication when trigin is asserted. **1** Enable trigger indication when trigin is asserted.
[7]	`0b0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[6]	`0b0`	FOnMan	Read-write	Flush Manual. Writing 1 to this bit generates a flush request on afvalid_s pin, writing 0 has no effect. It is automatically cleared when the generated flush request completes and afready_s is received by the TPIU. Reading this bit returns its current value.
[5]	`0b0`	FOnTrig	Read-write	Flush on Trigger Event. Initiates a flush request when a trigger event occurs. A trigger event occurs when the trigger counter reaches 0 while downcounting, or, if the TCVR is `0x0` and trigin goes HIGH. **0** Disable generation of flush when a trigger event occurs. **1** Enable generation of flush when a trigger event occurs.
[4]	`0b0`	FOnFlIn	Read-write	Flush on flushin. **0** Disable generation of flush using flushin input. **1** Enable generation of flush using flushin input.
[3:2]	`0b00`	RES0	Read-write	Reserved bit or field with SBZP behavior
[1]	`0b0`	EnFCont	Read-write	Enable Continuous Formatting Mode and Enable Formatter. The trigger packets are embedded in the trace stream. This bit can only be changed when FFSR.FtStopped is HIGH.
[0]	`0b0`	EnFTC	Read-write	Enable Formatter. The trigger packets are not embedded in the trace stream and the trace disable cycles and triggers are indicated by tracectl pin where present. This bit can only be changed when FFSR.FtStopped is HIGH.

10.12.2.11 css600_tpiu Formatter Synchronization Count Register, FSCR

The FSCR register indicates the maximum number of formatter frames sent to Trace Port after which a synchronization packet must be inserted. The register value indicates the programmed counter value and not the current state of the counter. The TPIU uses a frame sync counter that contains the number of formatter frames since the last frame synchronization packet. The counter is a 12-bit counter with a maximum count value of 4096. This equates to synchronization every 65536 bytes (4096 packets x 16 bytes per packet). On reset, the FSCR is set up for a synchronization packet every 1024 bytes, that is every 64 formatter frames. If the formatter is configured in continuous mode, full and half-word sync frames are inserted during normal operation. In this case, the counter value is the maximum number of complete frames between full synchronization packets. For the register access to complete on APB clocking on traceclk_in is needed.

Attributes

Offset

`0x0308`

Type

Read-write

Reset

`0x00000040`

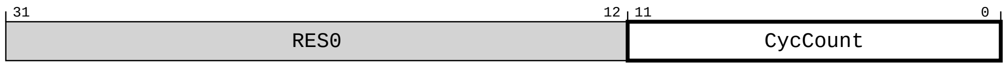
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-296: css600_tpiu_FSCR



The following table shows the bit assignments.

Table 10-307: FSCR attributes

Bits	Reset value	Name	Type	Function
[31:12]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[11:0]	`0x40`	CycCount	Read-write	12-bit counter value to indicate the number of complete frames between full synchronization packets. It is also used to send periodic synchronization requests to the ATB master using syncreq_s output. If this field is programmed as `0x0`, the synchronization counter is disabled. If this field is programmed with `0x1`-`0x7` the programmed value is `0x8`. The reset value is `0x040`, that is, 64 frames = 1024 bytes.

10.12.2.12 css600_tpiu External Control Port In Register, EXTCTLIN

Indicates the current status of external control input port extctl_in[7:0]. It can be used as a feedback mechanism for any serializers, pin sharing multiplexers, or other solutions that might be added to the trace output pins either for pin control or a high-speed trace port solution.

Attributes

Offset

`0x0400`

Type

Read-only

Reset

`0x000000--`

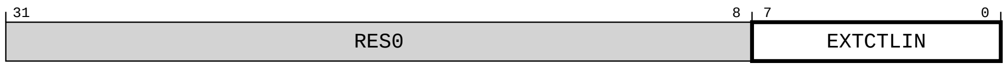
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-297: css600_tpiu_EXTCTLIN



The following table shows the bit assignments.

Table 10-308: EXTCTLIN attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	UNKNOWN	EXTCTLIN	Read-only	This 8-bit field shows the current status of external control input port extctl_in[7:0]. The reset value depends on the external source driving this port.

10.12.2.13 css600_tpiu External Control Port Out Register, EXTCTLOUT

Value to be driven on external control output port extctl_out[7:0]. It can be used as a control mechanism for any serializers, pin sharing multiplexers, or other solutions that might be added to the trace output pins either for pin control or a high-speed trace port solution.

Attributes

Offset

`0x0404`

Type

Read-write

Reset

`0x00000000`

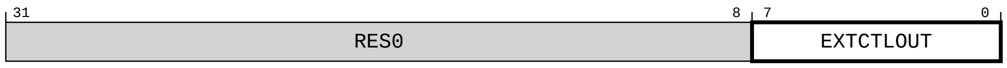
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-298: css600_tpiu_EXTCTLOUT



The following table shows the bit assignments.

Table 10-309: EXTCTLOUT attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	EXTCTLOUT	Read-write	This 8-bit field holds the value to be driven on the external control output port extctl_out[7:0].

10.12.2.14 css600_tpiu Integration Test Trigger In and Flush In Register, ITTRFLIN

This register indicates the integration status of the flushin and trigin inputs in integration mode. Reads are allowed even in functional mode, but the register itself is disabled and does not get updated even if the inputs change. The reset value depends on the external source driving the inputs.

Attributes

Offset

`0x0EE8`

Type

Read-only

Reset

`0x00000000`

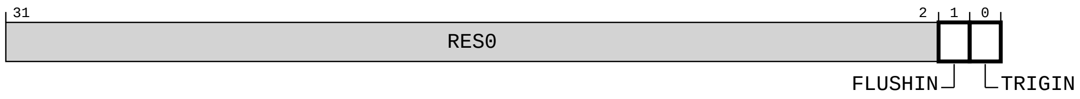
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-299: css600_tpiu_ITTRFLIN



The following table shows the bit assignments.

Table 10-310: ITTRFLIN attributes

Bits	Reset value	Name	Type	Function
[31:2]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[1]	`0b0`	FLUSHIN	Read-only	In integration mode, this bit latches to 1 on a rising edge of the flushin input. It is cleared when this register is read, or when integration mode is disabled.
[0]	`0b0`	TRIGIN	Read-only	In integration mode, this bit latches to 1 on a rising edge of the trigin input. It is cleared when this register is read or when integration mode is disabled.

10.12.2.15 css600_tpiu Integration Test ATB Data Register 0, ITATBDATA0

This register indicates the value of the atdata_s input in integration mode. Only 5 bits are readable through this register, the MSB of each of the four data bytes and the LSB. Reads are allowed even in functional mode, but the register is disabled and does not get updated even if the inputs change. The reset value depends on the external source driving the inputs.

Attributes

Offset

`0x0EEC`

Type

Read-only

Reset

`0x00000000`

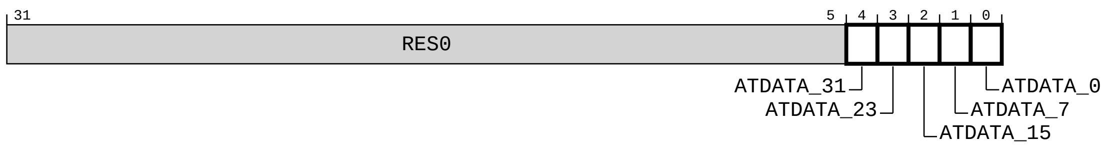
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-300: css600_tpiu_ITATBDATA0



The following table shows the bit assignments.

Table 10-311: ITATBDATA0 attributes

Bits	Reset value	Name	Type	Function
[31:5]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[4]	`0b0`	ATDATA_31	Read-only	Reads the value of atdata_s[31] during integration mode.
[3]	`0b0`	ATDATA_23	Read-only	Reads the value of atdata_s[23] during integration mode.
[2]	`0b0`	ATDATA_15	Read-only	Reads the value of atdata_s[15] during integration mode.
[1]	`0b0`	ATDATA_7	Read-only	Reads the value of atdata_s[7] during integration mode.
[0]	`0b0`	ATDATA_0	Read-only	Reads the value of atdata_s[0] during integration mode.

10.12.2.16 css600_tpiu Integration Test ATB Control Register 2, ITATBCTR2

This register enables control of the atready_s, afvalid_s, and syncreq_s outputs in integration mode. Writes to this register are allowed in integration mode as well as functional mode. However, the programmed value is driven to the outputs only in integration mode.

Attributes

Offset

`0x0EF0`

Type

Write-only

Reset

`0x00000000`

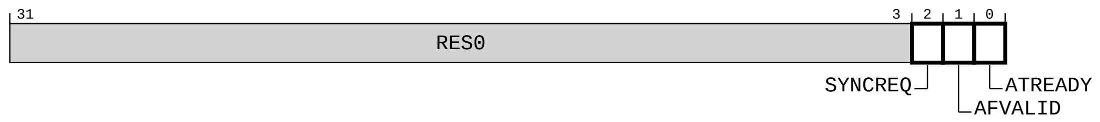
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-301: css600_tpiu_ITATBCTR2



The following table shows the bit assignments.

Table 10-312: ITATBCTR2 attributes

Bits	Reset value	Name	Type	Function
[31:3]	`0x0`	RES0	Write-only	Reserved bit or field with SBZP behavior
[2]	`0b0`	SYNCREQ	Write-only	Sets the value of syncreq_s in integration mode.
[1]	`0b0`	AFVALID	Write-only	Sets the value of afvalid_s in integration mode.

Bits	Reset value	Name	Type	Function
[0]	`0b0`	ATREADY	Write-only	Sets the value of atready_s in integration mode.

10.12.2.17 css600_tpiu Integration Test ATB Control Register 1, ITATBCTR1

This register indicates the value of the atid_s input in integration mode. Reads are allowed even in functional mode, but the register is disabled and does not get updated even if the inputs change. The reset value depends on external source driving these inputs.

Attributes

Offset

`0x0EF4`

Type

Read-only

Reset

`0x00000000`

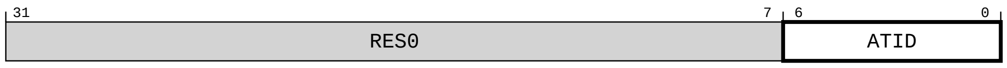
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-302: css600_tpiu_ITATBCTR1



The following table shows the bit assignments.

Table 10-313: ITATBCTR1 attributes

Bits	Reset value	Name	Type	Function
[31:7]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[6:0]	`0x0`	ATID	Read-only	Reads the value of atid_s[6:0] in integration mode.

10.12.2.18 css600_tpiu Integration Test ATB Control Register 0, ITATBCTR0

This register indicates the values of atvalid_s, afready_s, and atbytes_s inputs in integration mode. Reads are allowed even in functional mode, but the register is disabled and does not get updated even if the inputs change. The reset value depends on an external source driving these inputs.

Attributes

Offset

`0x0EF8`

Type

Read-only

Reset

`0x00000000`

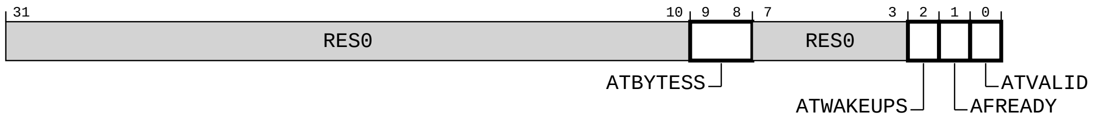
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-303: css600_tpiu_ITATBCTR0



The following table shows the bit assignments.

Table 10-314: ITATBCTR0 attributes

Bits	Reset value	Name	Type	Function
[31:10]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[9:8]	`0b00`	ATBYTESS	Read-only	Reads the value of atbytes_s[1:0] in integration mode.
[7:3]	`0b00000`	RES0	Read-only	Reserved bit or field with SBZP behavior
[2]	`0b0`	ATWAKEUPS	Read-only	Reads the value of atwakeup_s in integration mode.
[1]	`0b0`	AFREADY	Read-only	Reads the value of afready_s in integration mode.
[0]	`0b0`	ATVALID	Read-only	Reads the value of atvalid_s in integration mode.

10.12.2.19 css600_tpiu Integration Test Output Control Register, ITOUTCTR

This register enables control of the flushcomp output in integration mode. Writes to this register are allowed in integration mode, as well as functional mode. However, the programmed value is driven to the output pin only in integration mode.

Attributes

Offset

`0x0EFC`

Type

Write-only

Reset

`0x00000000`

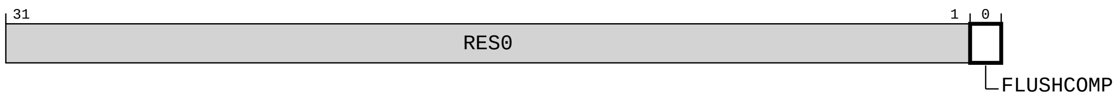
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-304: css600_tpiu_ITOUTCTR



The following table shows the bit assignments.

Table 10-315: ITOUTCTR attributes

Bits	Reset value	Name	Type	Function
[31:1]	`0x0`	RES0	Write-only	Reserved bit or field with SBZP behavior
[0]	`0b0`	FLUSHCOMP	Write-only	Sets the value of flushcomp in integration mode.

10.12.2.20 css600_tpiu Integration Mode Control Register, ITCTRL

The Integration Mode Control register is used to enable topology detection.

Attributes

Offset

`0x0F00`

Type

Read-write

Reset

`0x00000000`

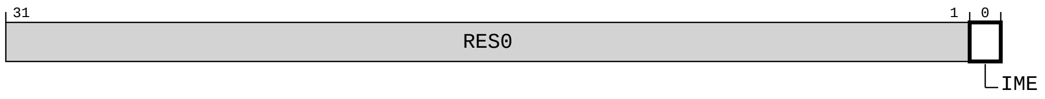
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-305: css600_tpiu_ITCTRL



The following table shows the bit assignments.

Table 10-316: ITCTRL attributes

Bits	Reset value	Name	Type	Function
[31:1]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[0]	`0b0`	IME	Read-write	Integration Mode Enable. When set, the component enters integration mode, enabling topology detection or integration testing to be performed.

10.12.2.21 css600_tpiu Claim Tag Set Register, CLAIMSET

This register forms one half of the claim tag value. On writes, this location enables individual bits to be set. On reads, it returns the number of bits that can be set.

Attributes

Offset

`0x0FA0`

Type

Read-write

Reset

`0x0000000F`

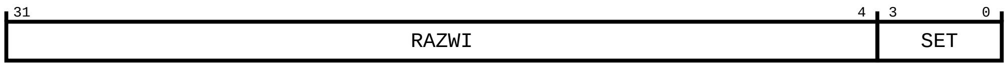
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-306: css600_tpiu_CLAIMSET



The following table shows the bit assignments.

Table 10-317: CLAIMSET attributes

Bits	Reset value	Name	Type	Function
[31:4]	`0x0`	RAZ/WI	Read-write	RAZ/WI
[3:0]	`0b1111`	SET	Read-write	A bit-programmable register bank that sets the claim tag value. A read returns a logic 1 for all implemented locations.

10.12.2.22 css600_tpiu Claim Tag Clear Register, CLAIMCLR

This register forms one half of the claim tag value. On writes, this location enables individual bits to be cleared. On reads, it returns the current claim tag value.

Attributes

Offset

`0x0FA4`

Type

Read-write

Reset

`0x00000000`

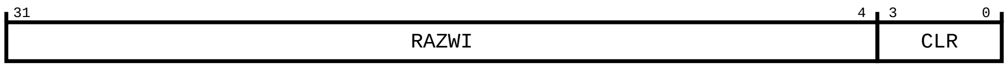
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-307: css600_tpiu_CLAIMCLR



The following table shows the bit assignments.

Table 10-318: CLAIMCLR attributes

Bits	Reset value	Name	Type	Function
[31:4]	`0x0`	RAZ/WI	Read-write	RAZ/WI
[3:0]	`0b0000`	CLR	Read-write	A bit-programmable register bank that clears the claim tag value. It is zero at reset. It is used by software agents to signal to each other ownership of the hardware. It has no direct effect on the hardware itself.

10.12.2.23 css600_tpiu Authentication Status Register, AUTHSTATUS

Reports the current status of the authentication control signals.

Attributes

Offset

`0x0FB8`

Type

Read-only

Reset

`0x00000000`

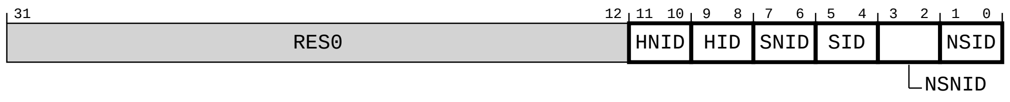
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-308: css600_tpiu_AUTHSTATUS



The following table shows the bit assignments.

Table 10-319: AUTHSTATUS attributes

Bits	Reset value	Name	Type	Function
[31:12]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[11:10]	`0b00`	HNID	Read-only	Hypervisor non-invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.

Bits	Reset value	Name	Type	Function
[9:8]	`0b00`	HID	Read-only	Hypervisor invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.
[7:6]	`0b00`	SNID	Read-only	Secure non-invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.
[5:4]	`0b00`	SID	Read-only	Secure invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.
[3:2]	`0b00`	NSNID	Read-only	Non-secure non-invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.
[1:0]	`0b00`	NSID	Read-only	Non-secure invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.

10.12.2.24 css600_tpiu Device Architecture Register, DEVARCH

Identifies the architect and architecture of a CoreSight component. The architect might differ from the designer of a component, for example Arm defines the architecture but another company designs and implements the component.

Attributes

Offset

`0x0FBC`

Type

Read-only

Reset

`0x00000000`

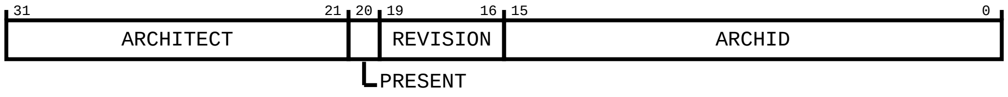
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-309: css600_tpiu_DEVARCH



The following table shows the bit assignments.

Table 10-320: DEVARCH attributes

Bits	Reset value	Name	Type	Function
[31:21]	`0x0`	ARCHITECT	Read-only	Returns 0.
[20]	`0b0`	PRESENT	Read-only	Returns 0, indicating that the DEVARCH register is not present.
[19:16]	`0b0000`	REVISION	Read-only	Returns 0
[15:0]	`0x0`	ARCHID	Read-only	Returns 0.

10.12.2.25 css600_tpiu Device Configuration Register, DEVID

The register indicates the capabilities of the component.

Attributes

Offset

`0x0FC8`

Type

Read-only

Reset

`0x00000020`

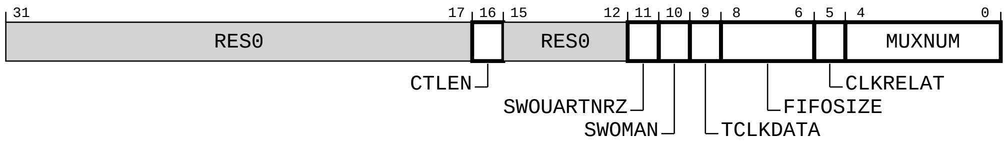
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-310: css600_tpiu_DEVID



The following table shows the bit assignments.

Table 10-321: DEVID attributes

Bits	Reset value	Name	Type	Function
[31:17]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[16]	`0b0`	CTLEN	Read-only	Trace Capture Enable support. Reads `0x1`, which indicates that the CTL register is implemented and the software can enable and disable trace capture by programming the CTL register.
[15:12]	`0b0000`	RES0	Read-only	Reserved bit or field with SBZP behavior
[11]	`0b0`	SWOUARTNRZ	Read-only	Serial Wire Output, UART or NRZ support. Reads `0x0`, which indicates that Serial Wire Output, UART or NRZ, is not supported.
[10]	`0b0`	SWOMAN	Read-only	Serial Wire Output, Manchester-encoded format support. Reads `0x0`, which indicates that Serial Wire Output, Manchester-encoded format, is not supported.
[9]	`0b0`	TCLKDATA	Read-only	Trace Clock Plus Data support. Reads `0x0`, which indicates that trace clock and data is supported.
[8:6]	`0b000`	FIFOSIZE	Read-only	FIFO size in powers of 2. Reads `0x0`, indicating that the FIFO size is implementation-defined and is not visible in the programmers model.
[5]	`0b1`	CLKRELAT	Read-only	Relationship between clk and traceclk_in. Reads `0x1` which indicates that these two clocks are asynchronous.
[4:0]	`0b00000`	MUXNUM	Read-only	Indicates a hidden level of input multiplexing. When non-zero, this value indicates the type of multiplexing on the input to the ATB. Currently only `0x00` is supported, that is, no multiplexing is present. This value helps detect the ATB structure.

10.12.2.26 css600_tpiu Device Type Identifier Register, DEVTYPE

A debugger can use this register to get information about a component that has an unrecognized Part number.

Attributes

Offset

`0x0FCC`

Type

Read-only

Reset

`0x00000011`

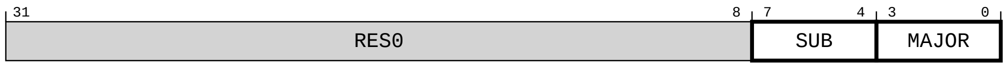
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-311: css600_tpiu_DEVTYPE



The following table shows the bit assignments.

Table 10-322: DEVTYPE attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0001`	SUB	Read-only	Minor classification. Returns `0x1`, indicating this component is a Trace Port.
[3:0]	`0b0001`	MAJOR	Read-only	Major classification. Returns `0x1`, indicating this component is a Trace Sink.

10.12.2.27 css600_tpiu Peripheral Identification Register 4, PIDR4

The PIDR4 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FD0`

Type

Read-only

Reset

`0x00000004`

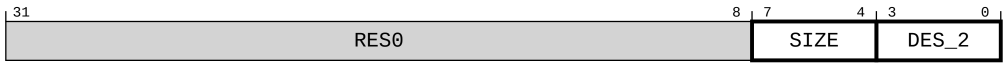
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-312: css600_tpiu_PIDR4



The following table shows the bit assignments.

Table 10-323: PIDR4 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior

Bits	Reset value	Name	Type	Function
[7:4]	`0b0000`	SIZE	Read-only	Indicates the memory size that is used by this component. Returns 0 indicating that the component uses an **UNKNOWN** number of 4KB blocks. Using the SIZE field to indicate the size of the component is deprecated.
[3:0]	`0b0100`	DES_2	Read-only	JEP106 continuation code. Together, with PIDR2.DES_1 and PIDR1.DES_0, they indicate the designer of the component and not the implementer, except where the two are the same.

10.12.2.28 css600_tpiu Peripheral Identification Register 5, PIDR5

The PIDR5 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FD4`

Type

Read-only

Reset

`0x00000000`

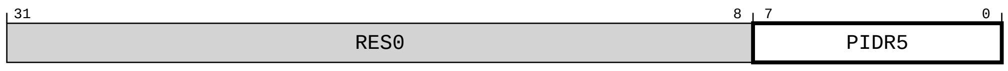
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-313: css600_tpiu_PIDR5



The following table shows the bit assignments.

Table 10-324: PIDR5 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	PIDR5	Read-only	Reserved.

10.12.2.29 css600_tpiu Peripheral Identification Register 6, PIDR6

The PIDR6 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FD8`

Type

Read-only

Reset

`0x00000000`

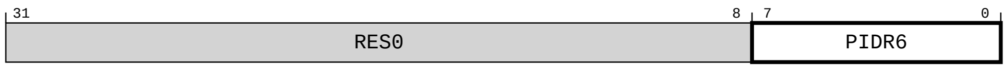
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-314: css600_tpiu_PIDR6



The following table shows the bit assignments.

Table 10-325: PIDR6 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	PIDR6	Read-only	Reserved.

10.12.2.30 css600_tpiu Peripheral Identification Register 7, PIDR7

The PIDR7 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FDC`

Type

Read-only

Reset

`0x00000000`

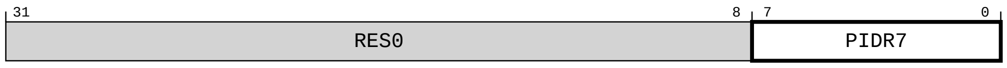
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-315: css600_tpiu_PIDR7



The following table shows the bit assignments.

Table 10-326: PIDR7 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	PIDR7	Read-only	Reserved.

10.12.2.31 css600_tpiu Peripheral Identification Register 0, PIDR0

The PIDR0 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FE0`

Type

Read-only

Reset

`0x000000E7`

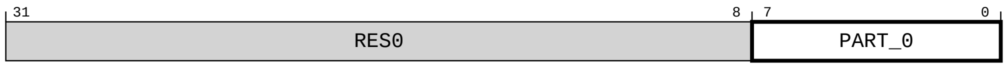
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-316: css600_tpiu_PIDR0



The following table shows the bit assignments.

Table 10-327: PIDR0 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	0xE7	PART_0	Read-only	Part number, bits[7:0]. Taken together with PIDR1.PART_1 it indicates the component. The Part Number is selected by the designer of the component.

10.12.2.32 css600_tpiu Peripheral Identification Register 1, PIDR1

The PIDR1 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FE4`

Type

Read-only

Reset

`0x000000B9`

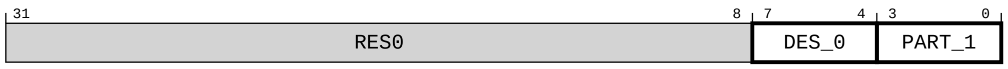
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-317: css600_tpiu_PIDR1



The following table shows the bit assignments.

Table 10-328: PIDR1 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b1011`	DES_0	Read-only	JEP106 identification code, bits[3:0]. Together, with PIDR4.DES_2 and PIDR2.DES_1, they indicate the designer of the component and not the implementer, except where the two are the same.
[3:0]	`0b1001`	PART_1	Read-only	Part number, bits[11:8]. Taken together with PIDR0.PART_0 it indicates the component. The Part Number is selected by the designer of the component.

10.12.2.33 css600_tpiu Peripheral Identification Register 2, PIDR2

The PIDR2 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FE8`

Type

Read-only

Reset

`0x0000002B`

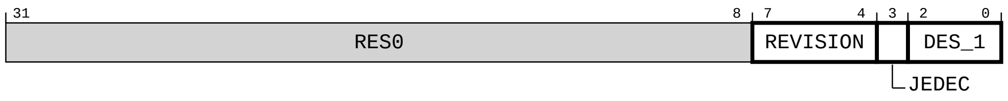
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-318: css600_tpiu_PIDR2



The following table shows the bit assignments.

Table 10-329: PIDR2 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0010`	REVISION	Read-only	Revision. It is an incremental value starting at `0x0` for the first design of a component. See the Component list in Chapter 1 for information on the RTL revision of the component.
[3]	`0b1`	JEDEC	Read-only	1 - Always set. Indicates that a JEDEC assigned value is used.
[2:0]	`0b011`	DES_1	Read-only	JEP106 identification code, bits[6:4]. Together, with PIDR4.DES_2 and PIDR1.DES_0, they indicate the designer of the component and not the implementer, except where the two are the same.

10.12.2.34 css600_tpiu Peripheral Identification Register 3, PIDR3

The PIDR3 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FEC`

Type

Read-only

Reset

`0x00000000`

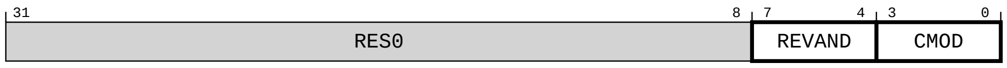
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-319: css600_tpiu_PIDR3



The following table shows the bit assignments.

Table 10-330: PIDR3 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0000`	REVAND	Read-only	This field indicates minor errata fixes specific to this design, for example metal fixes after implementation. In most cases this field is `0x0`.
[3:0]	`0b0000`	CMOD	Read-only	Customer Modified. Where the component is reusable IP, this value indicates if the customer has modified the behavior of the component. In most cases this field is `0x0`.

10.12.2.35 css600_tpiu Component Identification Register 0, CIDR0

The CIDR0 register is part of the set of component identification registers.

Attributes

Offset

`0x0FF0`

Type

Read-only

Reset

`0x0000000D`

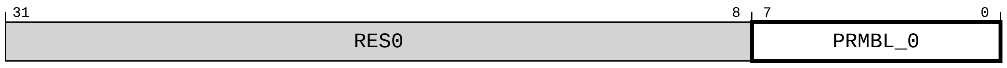
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-320: css600_tpiu_CIDR0



The following table shows the bit assignments.

Table 10-331: CIDR0 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0xD`	PRMBL_0	Read-only	Preamble. Returns `0x0D`.

10.12.2.36 css600_tpiu Component Identification Register 1, CIDR1

The CIDR1 register is part of the set of component identification registers.

Attributes

Offset

`0x0FF4`

Type

Read-only

Reset

`0x00000090`

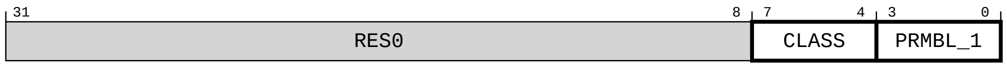
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-321: css600_tpiu_CIDR1



The following table shows the bit assignments.

Table 10-332: CIDR1 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b1001`	CLASS	Read-only	Component class. Returns `0x9`, indicating this is a CoreSight component.
[3:0]	`0b0000`	PRMBL_1	Read-only	Preamble. Returns `0x0`.

10.12.2.37 css600_tpiu Component Identification Register 2, CIDR2

The CIDR2 register is part of the set of component identification registers.

Attributes

Offset

`0x0FF8`

Type

Read-only

Reset

`0x00000005`

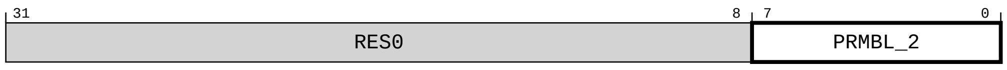
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-322: css600_tpiu_CIDR2



The following table shows the bit assignments.

Table 10-333: CIDR2 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x5`	PRMBL_2	Read-only	Preamble. Returns `0x05`.

10.12.2.38 css600_tpiu Component Identification Register 3, CIDR3

The CIDR3 register is part of the set of component identification registers.

Attributes

Offset
`0x0FFC`

Type
Read-only

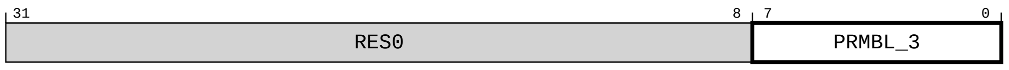
Reset
`0x000000B1`

Width
32

Bit descriptions

The following image shows the bit assignments.

Figure 10-323: css600_tpiu_CIDR3



The following table shows the bit assignments.

Table 10-334: CIDR3 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0xB1`	PRMBL_3	Read-only	Preamble. Returns `0xB1`.

10.13 `css600_tsgen` introduction

This section describes the programmers model of the `css600_tsgen`.

Note

The css600_tsgen has two interfaces: a control interface, referred to in the following sections as APB4_Slave_0, and a read-only interface, referred to as APB4_Slave_1.

10.13.1 APB4_Slave_0 register summary

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

Note

A reset value containing one or more '-' means that this register contains *unknown* or *implementation-defined* values. See the relevant register description for more information.

Locations that are not listed in the table are Reserved.

Table 10-335: css600_tsgen_apb4_slave_0 register summary

Offset	Name	Type	Reset	Width	Description
0x0000	CNTCR	RW	0x00000000	32	css600_tsgen_apb4_slave_0 Counter Control Register, CNTCR
0x0004	CNTSR	RO	0x00000000	32	css600_tsgen_apb4_slave_0 Counter Status Register, CNTSR
0x0008	CNTCVL	RW	0x00000000	32	css600_tsgen_apb4_slave_0 Current value of Counter[31:0], CNTCVL
0x000C	CNTCVU	RW	0x00000000	32	css600_tsgen_apb4_slave_0 Current value of Counter[63:32], CNTCVU
0x0020	CNTFID0	RW	0x00000000	32	css600_tsgen_apb4_slave_0 Base Frequency ID register, CNTFID0
0x0EF8	ITSTAT	RO	0x0000000-	32	css600_tsgen_apb4_slave_0 Integration Test Status Register, ITSTAT
0x0F00	ITCTRL	RW	0x00000000	32	css600_tsgen_apb4_slave_0 Integration Mode Control Register, ITCTRL
0x0FD0	PIDR4	RO	0x00000004	32	css600_tsgen_apb4_slave_0 Peripheral Identification Register 4, PIDR4
0x0FD4	PIDR5	RO	0x00000000	32	css600_tsgen_apb4_slave_0 Peripheral Identification Register 5, PIDR5
0x0FD8	PIDR6	RO	0x00000000	32	css600_tsgen_apb4_slave_0 Peripheral Identification Register 6, PIDR6
0x0FDC	PIDR7	RO	0x00000000	32	css600_tsgen_apb4_slave_0 Peripheral Identification Register 7, PIDR7
0x0FE0	PIDR0	RO	0x00000093	32	css600_tsgen_apb4_slave_0 Peripheral Identification Register 0, PIDR0
0x0FE4	PIDR1	RO	0x000000B1	32	css600_tsgen_apb4_slave_0 Peripheral Identification Register 1, PIDR1
0x0FE8	PIDR2	RO	0x0000000B	32	css600_tsgen_apb4_slave_0 Peripheral Identification Register 2, PIDR2
0x0FEC	PIDR3	RO	0x00000000	32	css600_tsgen_apb4_slave_0 Peripheral Identification Register 3, PIDR3
0x0FF0	CIDR0	RO	0x0000000D	32	css600_tsgen_apb4_slave_0 Component Identification Register 0, CIDR0
0x0FF4	CIDR1	RO	0x000000F0	32	css600_tsgen_apb4_slave_0 Component Identification Register 1, CIDR1
0x0FF8	CIDR2	RO	0x00000005	32	css600_tsgen_apb4_slave_0 Component Identification Register 2, CIDR2
0x0FFC	CIDR3	RO	0x000000B1	32	css600_tsgen_apb4_slave_0 Component Identification Register 3, CIDR3

10.13.2 APB4_Slave_0 register descriptions

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

APB4_Slave_0 register summary provides cross references to individual registers.

APB4_Slave_1 register summary provides cross references to individual registers.

10.13.2.1 css600_tsgen_apb4_slave_0 Counter Control Register, CNTCR

The CNTCR register controls the counter increments.

Attributes

Offset

`0x0000`

Type

Read-write

Reset

`0x00000000`

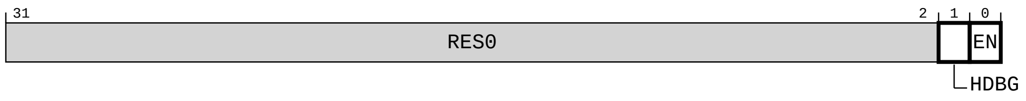
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-324: css600_tsgen_apb4_slave_0_CNTCR



The following table shows the bit assignments.

Table 10-336: CNTCR attributes

Bits	Reset value	Name	Type	Function
[31:2]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[1]	`0b0`	HDBG	Read-write	Halt On Debug: **0** Do not halt on debug. The halt_req signal into the counter has no effect. **1** Halt on debug. When the halt_req pulse is received, the count value is held static.
[0]	`0b0`	EN	Read-write	Enable Bit. **0** The counter is disabled. Count is not incrementing. **1** The counter is enabled. Count is incrementing.

10.13.2.2 css600_tsgen_apb4_slave_0 Counter Status Register, CNTSR

The CNTSR register identifies the status of the counter.

Attributes

Offset

`0x0004`

Type

Read-only

Reset

`0x00000000`

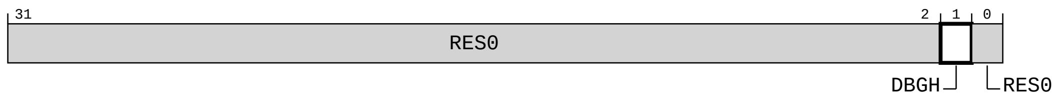
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-325: css600_tsgen_apb4_slave_0_CNTSR



The following table shows the bit assignments.

Table 10-337: CNTSR attributes

Bits	Reset value	Name	Type	Function
[31:2]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[1]	`0b0`	DBGH	Read-only	Debug status: **0** Debug is halted **1** Debug is not halted
[0]	`0b0`	RES0	Read-only	Reserved bit or field with SBZP behavior

10.13.2.3 css600_tsgen_apb4_slave_0 Current value of Counter[31:0], CNTCVL

The CNTCVL register reads or writes the lower 32 bits of the current counter value

Attributes

Offset

`0x0008`

Type

Read-write

Reset

`0x00000000`

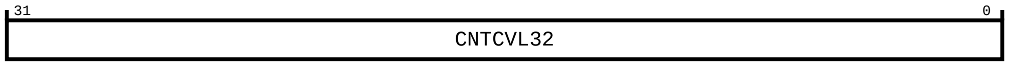
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-326: css600_tsgen_apb4_slave_0_CNTCVL



The following table shows the bit assignments.

Table 10-338: CNTCVL attributes

Bits	Reset value	Name	Type	Function
[31:0]	`0x0`	CNTCVL32	Read-write	Reads to this register return the lower 32 bits of the current timestamp counter value. To change the current timestamp value, write the lower 32 bits of the new value to this register before writing the upper 32 bits to CNTCVU. The timestamp value is not changed until the CNTCVU register is written to.

10.13.2.4 css600_tsgen_apb4_slave_0 Current value of Counter[63:32], CNTCVU

The CNTCVU register reads or writes the upper 32 bits of the current counter value. The control interface must clear the CNTCR.EN bit or set CNTCR.HDBG and hlt_dbg asserted on the input to stop the counter, before writing to this register.

Attributes

Offset

`0x000C`

Type

Read-write

Reset

`0x00000000`

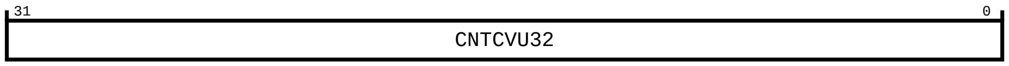
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-327: css600_tsgen_apb4_slave_0_CNTCVU



The following table shows the bit assignments.

Table 10-339: CNTCVU attributes

Bits	Reset value	Name	Type	Function
[31:0]	`0x0`	CNTCVU32	Read-write	Reads to this register return the upper 32 bits of the current timestamp counter value. To change the current timestamp value, write the lower 32 bits of the new value to CNTCVL before writing the upper 32 bits to this register. The 64-bit timestamp value is updated with the value from both writes when this register is written to.

10.13.2.5 css600_tsgen_apb4_slave_0 Base Frequency ID register, CNTFID0

You must program the CNTFID0 register to match the clock frequency of the timestamp generator, in ticks per second. For example, for a 50 MHz clock, program 0x02FAF080. The real-time speed of the counter does not depend on the value of this register. This register reports, to the reader, the speed of the counter as programmed by the system firmware.

Attributes

Offset

`0x0020`

Type

Read-write

Reset

`0x00000000`

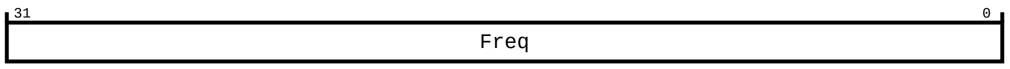
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-328: css600_tsgen_apb4_slave_0_CNTFID0



The following table shows the bit assignments.

Table 10-340: CNTFID0 attributes

Bits	Reset value	Name	Type	Function
[31:0]	`0x0`	Freq	Read-write	Frequency in number of ticks per second. Up to 4GHz can be specified.

10.13.2.6 css600_tsgen_apb4_slave_0 Integration Test Status Register, ITSTAT

The ITSTAT register views the halt_req and restart_req values.

Attributes

Offset

`0x0EF8`

Type

Read-only

Reset

`0x0000000-`

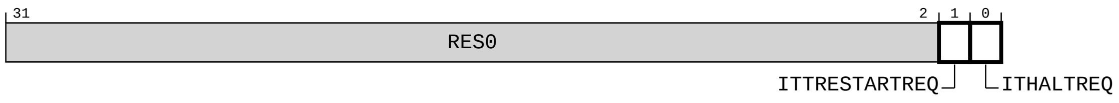
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-329: css600_tsgen_apb4_slave_0_ITSTAT



The following table shows the bit assignments.

Table 10-341: ITSTAT attributes

Bits	Reset value	Name	Type	Function
[31:2]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[1]	UNKNOWN	ITTRESTARTREQ	Read-only	Integration Test Restart Request status of the restart_req input. Integration testing mode: Behaves as a sticky bit and latches to 1 when tsgen receives restart request. Cleared on reading this register. If restart_req is asserted in the same cycle as an APB read of this register, the read takes priority and the register is cleared as a result. Always returns 0 in normal functional mode.
[0]	UNKNOWN	ITHALTREQ	Read-only	Integration Test Halt Request status of the halt_req input. Integration testing mode: Behaves as a sticky bit and latches to 1 when tsgen receives halt request. Cleared on reading this register. If halt_req is asserted in the same cycle as an APB read of this register, the read takes priority and the register is cleared as a result. Always returns 0 in normal functional mode.

10.13.2.7 css600_tsgen_apb4_slave_0 Integration Mode Control Register, ITCTRL

The Integration Mode Control register is used to enable topology detection.

Attributes

Offset

`0x0F00`

Type

Read-write

Reset

`0x00000000`

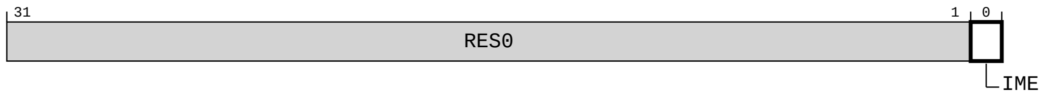
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-330: css600_tsgen_apb4_slave_0_ITCTRL



The following table shows the bit assignments.

Table 10-342: ITCTRL attributes

Bits	Reset value	Name	Type	Function
[31:1]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[0]	`0b0`	IME	Read-write	Integration Mode Enable. When set, the component enters integration mode, enabling topology detection or integration testing to be performed.

10.13.2.8 css600_tsgen_apb4_slave_0 Peripheral Identification Register 4, PIDR4

The PIDR4 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FD0`

Type

Read-only

Reset

`0x00000004`

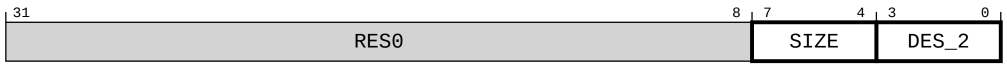
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-331: css600_tsgen_apb4_slave_0_PIDR4



The following table shows the bit assignments.

Table 10-343: PIDR4 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0000`	SIZE	Read-only	Indicates the memory size that is used by this component. Returns 0 indicating that the component uses an **UNKNOWN** number of 4KB blocks. Using the SIZE field to indicate the size of the component is deprecated.
[3:0]	`0b0100`	DES_2	Read-only	JEP106 continuation code. Together, with PIDR2.DES_1 and PIDR1.DES_0, they indicate the designer of the component and not the implementer, except where the two are the same.

10.13.2.9 css600_tsgen_apb4_slave_0 Peripheral Identification Register 5, PIDR5

The PIDR5 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FD4`

Type

Read-only

Reset

`0x00000000`

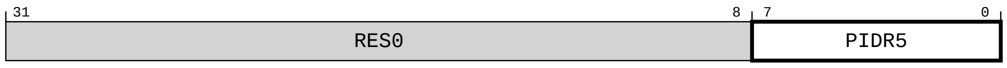
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-332: css600_tsgen_apb4_slave_0_PIDR5



The following table shows the bit assignments.

Table 10-344: PIDR5 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	PIDR5	Read-only	Reserved.

10.13.2.10 css600_tsgen_apb4_slave_0 Peripheral Identification Register 6, PIDR6

The PIDR6 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FD8`

Type

Read-only

Reset

`0x00000000`

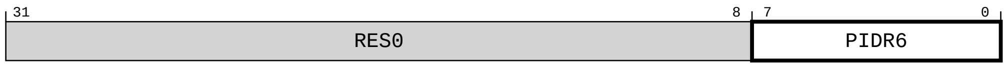
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-333: css600_tsgen_apb4_slave_0_PIDR6



The following table shows the bit assignments.

Table 10-345: PIDR6 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	PIDR6	Read-only	Reserved.

10.13.2.11 css600_tsgen_apb4_slave_0 Peripheral Identification Register 7, PIDR7

The PIDR7 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FDC`

Type

Read-only

Reset

`0x00000000`

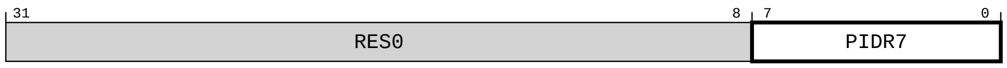
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-334: css600_tsgen_apb4_slave_0_PIDR7



The following table shows the bit assignments.

Table 10-346: PIDR7 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	PIDR7	Read-only	Reserved.

10.13.2.12 css600_tsgen_apb4_slave_0 Peripheral Identification Register 0, PIDR0

The PIDR0 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FE0`

Type

Read-only

Reset

`0x00000093`

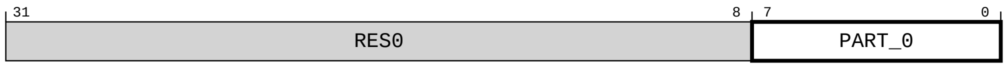
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-335: css600_tsgen_apb4_slave_0_PIDR0



The following table shows the bit assignments.

Table 10-347: PIDR0 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x93`	PART_0	Read-only	Part number, bits[7:0]. Taken together with PIDR1.PART_1 it indicates the component. The Part Number is selected by the designer of the component.

10.13.2.13 css600_tsgen_apb4_slave_0 Peripheral Identification Register 1, PIDR1

The PIDR1 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FE4`

Type

Read-only

Reset

`0x000000B1`

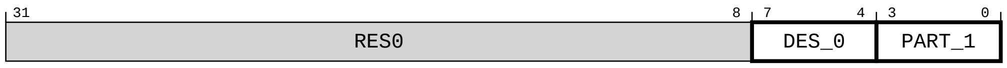
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-336: css600_tsgen_apb4_slave_0_PIDR1



The following table shows the bit assignments.

Table 10-348: PIDR1 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b1011`	DES_0	Read-only	JEP106 identification code, bits[3:0]. Together, with PIDR4.DES_2 and PIDR2.DES_1, they indicate the designer of the component and not the implementer, except where the two are the same.
[3:0]	`0b0001`	PART_1	Read-only	Part number, bits[11:8]. Taken together with PIDR0.PART_0 it indicates the component. The Part Number is selected by the designer of the component.

10.13.2.14 css600_tsgen_apb4_slave_0 Peripheral Identification Register 2, PIDR2

The PIDR2 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FE8`

Type

Read-only

Reset

`0x0000000B`

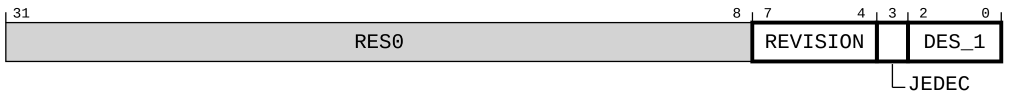
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-337: css600_tsgen_apb4_slave_0_PIDR2



The following table shows the bit assignments.

Table 10-349: PIDR2 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0000`	REVISION	Read-only	Revision. It is an incremental value starting at `0x0` for the first design of a component. See the Component list in Chapter 1 for information on the RTL revision of the component.

Bits	Reset value	Name	Type	Function
[3]	`0b1`	JEDEC	Read-only	1 - Always set. Indicates that a JEDEC assigned value is used.
[2:0]	`0b011`	DES_1	Read-only	JEP106 identification code, bits[6:4]. Together, with PIDR4.DES_2 and PIDR1.DES_0, they indicate the designer of the component and not the implementer, except where the two are the same.

10.13.2.15 css600_tsgen_apb4_slave_0 Peripheral Identification Register 3, PIDR3

The PIDR3 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FEC`

Type

Read-only

Reset

`0x00000000`

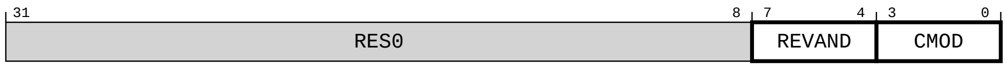
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-338: css600_tsgen_apb4_slave_0_PIDR3



The following table shows the bit assignments.

Table 10-350: PIDR3 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0000`	REVAND	Read-only	This field indicates minor errata fixes specific to this design, for example metal fixes after implementation. In most cases this field is `0x0`.
[3:0]	`0b0000`	CMOD	Read-only	Customer Modified. Where the component is reusable IP, this value indicates if the customer has modified the behavior of the component. In most cases this field is `0x0`.

10.13.2.16 css600_tsgen_apb4_slave_0 Component Identification Register 0, CIDR0

The CIDR0 register is part of the set of component identification registers.

Attributes

Offset

`0x0FF0`

Type

Read-only

Reset

`0x0000000D`

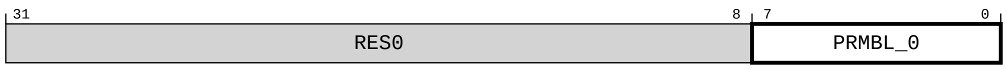
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-339: css600_tsgen_apb4_slave_0_CIDR0



The following table shows the bit assignments.

Table 10-351: CIDR0 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0xD`	PRMBL_0	Read-only	Preamble. Returns `0x0D`.

10.13.2.17 css600_tsgen_apb4_slave_0 Component Identification Register 1, CIDR1

The CIDR1 register is part of the set of component identification registers.

Attributes

Offset

`0x0FF4`

Type

Read-only

Reset

`0x000000F0`

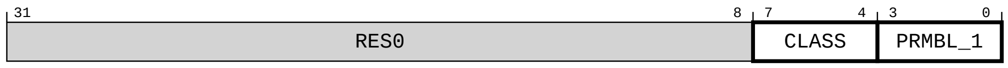
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-340: css600_tsgen_apb4_slave_0_CIDR1



The following table shows the bit assignments.

Table 10-352: CIDR1 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b1111`	CLASS	Read-only	Component class, returns 0xf, indicating CoreLink, PrimeCell, or system component
[3:0]	`0b0000`	PRMBL_1	Read-only	Preamble, returns `0x0`

10.13.2.18 css600_tsgen_apb4_slave_0 Component Identification Register 2, CIDR2

The CIDR2 register is part of the set of component identification registers.

Attributes

Offset

`0x0FF8`

Type

Read-only

Reset

`0x00000005`

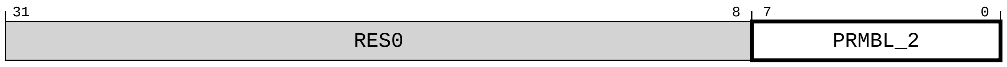
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-341: css600_tsgen_apb4_slave_0_CIDR2



The following table shows the bit assignments.

Table 10-353: CIDR2 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x5`	PRMBL_2	Read-only	Preamble. Returns `0x05`.

10.13.2.19 css600_tsgen_apb4_slave_0 Component Identification Register 3, CIDR3

The CIDR3 register is part of the set of component identification registers.

Attributes

Offset

`0x0FFC`

Type

Read-only

Reset

`0x000000B1`

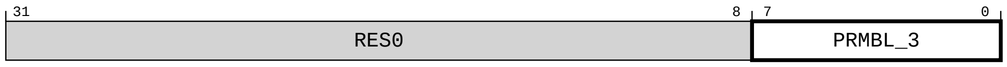
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-342: css600_tsgen_apb4_slave_0_CIDR3



The following table shows the bit assignments.

Table 10-354: CIDR3 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0xB1`	PRMBL_3	Read-only	Preamble. Returns `0xB1`.

10.13.3 APB4_Slave_1 register summary

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

Note

A reset value containing one or more '-' means that this register contains *unknown* or *implementation-defined* values. See the relevant register description for more information.

Locations that are not listed in the table are Reserved.

Table 10-355: css600_tsgen_apb4_slave_1 register summary

Offset	Name	Type	Reset	Width	Description
0x0008	CNTCVL	RW	0x00000000	32	css600_tsgen_apb4_slave_1 Current value of Counter[31:0], CNTCVLREAD
0x000C	CNTCVU	RW	0x00000000	32	css600_tsgen_apb4_slave_1 Current value of Counter[63:32], CNTCVUREAD
0x0FD0	PIDR4	RO	0x00000004	32	css600_tsgen_apb4_slave_1 Peripheral Identification Register 4, PIDR4
0x0FD4	PIDR5	RO	0x00000000	32	css600_tsgen_apb4_slave_1 Peripheral Identification Register 5, PIDR5
0x0FD8	PIDR6	RO	0x00000000	32	css600_tsgen_apb4_slave_1 Peripheral Identification Register 6, PIDR6
0x0FDC	PIDR7	RO	0x00000000	32	css600_tsgen_apb4_slave_1 Peripheral Identification Register 7, PIDR7
0x0FE0	PIDR0	RO	0x00000093	32	css600_tsgen_apb4_slave_1 Peripheral Identification Register 0, PIDR0
0x0FE4	PIDR1	RO	0x000000B1	32	css600_tsgen_apb4_slave_1 Peripheral Identification Register 1, PIDR1
0x0FE8	PIDR2	RO	0x0000000B	32	css600_tsgen_apb4_slave_1 Peripheral Identification Register 2, PIDR2
0x0FEC	PIDR3	RO	0x00000000	32	css600_tsgen_apb4_slave_1 Peripheral Identification Register 3, PIDR3
0x0FF0	CIDR0	RO	0x0000000D	32	css600_tsgen_apb4_slave_1 Component Identification Register 0, CIDR0
0x0FF4	CIDR1	RO	0x000000F0	32	css600_tsgen_apb4_slave_1 Component Identification Register 1, CIDR1
0x0FF8	CIDR2	RO	0x00000005	32	css600_tsgen_apb4_slave_1 Component Identification Register 2, CIDR2
0x0FFC	CIDR3	RO	0x000000B1	32	css600_tsgen_apb4_slave_1 Component Identification Register 3, CIDR3

10.13.4 APB4_Slave_1 register descriptions

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

APB4_Slave_0 register summary provides cross references to individual registers.

APB4_Slave_1 register summary provides cross references to individual registers.

10.13.4.1 css600_tsgen_apb4_slave_1 Current value of Counter[31:0], CNTCVLREAD

Reads or writes the lower 32 bits of the current counter value

Attributes

Offset

`0x0008`

Type

Read-write

Reset

`0x00000000`

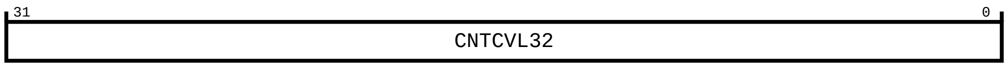
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-343: css600_tsgen_apb4_slave_1_CNTCVL



The following table shows the bit assignments.

Table 10-356: CNTCVL attributes

Bits	Reset value	Name	Type	Function
[31:0]	`0x0`	CNTCVL32	Read-write	The lower 32 bits of the current timestamp counter value.

10.13.4.2 css600_tsgen_apb4_slave_1 Current value of Counter[63:32], CNTCVUREAD

Reads the upper 32 bits of the current counter value.

Attributes

Offset

`0x000C`

Type

Read-write

Reset

`0x00000000`

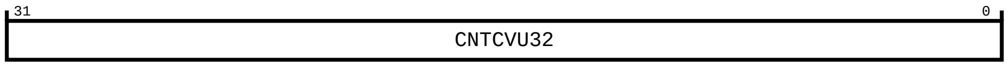
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-344: css600_tsgen_apb4_slave_1_CNTCVU



The following table shows the bit assignments.

Table 10-357: CNTCVU attributes

Bits	Reset value	Name	Type	Function
[31:0]	`0x0`	CNTCVU32	Read-write	The upper 32 bits of the current timestamp counter value.

10.13.4.3 css600_tsgen_apb4_slave_1 Peripheral Identification Register 4, PIDR4

The PIDR4 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FD0`

Type

Read-only

Reset

`0x00000004`

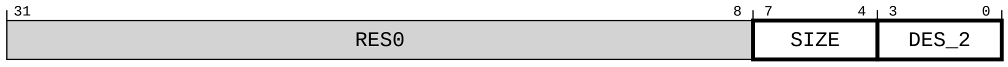
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-345: css600_tsgen_apb4_slave_1_PIDR4



The following table shows the bit assignments.

Table 10-358: PIDR4 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0000`	SIZE	Read-only	Indicates the memory size that is used by this component. Returns 0 indicating that the component uses an **UNKNOWN** number of 4KB blocks. Using the SIZE field to indicate the size of the component is deprecated.
[3:0]	`0b0100`	DES_2	Read-only	JEP106 continuation code. Together, with PIDR2.DES_1 and PIDR1.DES_0, they indicate the designer of the component and not the implementer, except where the two are the same.

10.13.4.4 css600_tsgen_apb4_slave_1 Peripheral Identification Register 5, PIDR5

The PIDR5 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FD4`

Type

Read-only

Reset

`0x00000000`

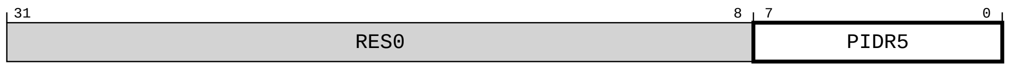
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-346: css600_tsgen_apb4_slave_1_PIDR5



The following table shows the bit assignments.

Table 10-359: PIDR5 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	PIDR5	Read-only	Reserved.

10.13.4.5 css600_tsgen_apb4_slave_1 Peripheral Identification Register 6, PIDR6

The PIDR6 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FD8`

Type

Read-only

Reset

`0x00000000`

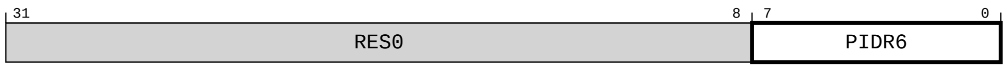
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-347: css600_tsgen_apb4_slave_1_PIDR6



The following table shows the bit assignments.

Table 10-360: PIDR6 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	PIDR6	Read-only	Reserved.

10.13.4.6 css600_tsgen_apb4_slave_1 Peripheral Identification Register 7, PIDR7

The PIDR7 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FDC`

Type

Read-only

Reset

`0x00000000`

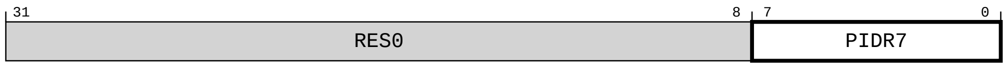
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-348: css600_tsgen_apb4_slave_1_PIDR7



The following table shows the bit assignments.

Table 10-361: PIDR7 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	PIDR7	Read-only	Reserved.

10.13.4.7 css600_tsgen_apb4_slave_1 Peripheral Identification Register 0, PIDR0

The PIDR0 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FE0`

Type

Read-only

Reset

`0x00000093`

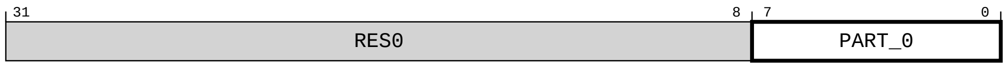
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-349: css600_tsgen_apb4_slave_1_PIDR0



The following table shows the bit assignments.

Table 10-362: PIDR0 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x93`	PART_0	Read-only	Part number, bits[7:0]. Taken together with PIDR1.PART_1 it indicates the component. The Part Number is selected by the designer of the component.

10.13.4.8 css600_tsgen_apb4_slave_1 Peripheral Identification Register 1, PIDR1

The PIDR1 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FE4`

Type

Read-only

Reset

`0x000000B1`

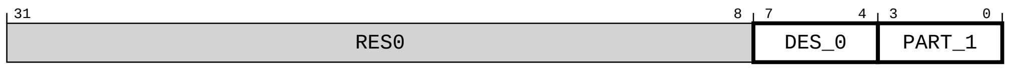
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-350: css600_tsgen_apb4_slave_1_PIDR1



The following table shows the bit assignments.

Table 10-363: PIDR1 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b1011`	DES_0	Read-only	JEP106 identification code, bits[3:0]. Together, with PIDR4.DES_2 and PIDR2.DES_1, they indicate the designer of the component and not the implementer, except where the two are the same.
[3:0]	`0b0001`	PART_1	Read-only	Part number, bits[11:8]. Taken together with PIDR0.PART_0 it indicates the component. The Part Number is selected by the designer of the component.

10.13.4.9 css600_tsgen_apb4_slave_1 Peripheral Identification Register 2, PIDR2

The PIDR2 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FE8`

Type

Read-only

Reset

`0x0000000B`

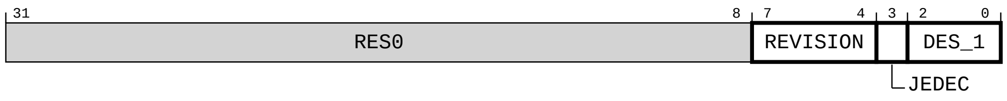
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-351: css600_tsgen_apb4_slave_1_PIDR2



The following table shows the bit assignments.

Table 10-364: PIDR2 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0000`	REVISION	Read-only	Revision. It is an incremental value starting at `0x0` for the first design of a component. See the Component list in Chapter 1 for information on the RTL revision of the component.
[3]	`0b1`	JEDEC	Read-only	1 - Always set. Indicates that a JEDEC assigned value is used.
[2:0]	`0b011`	DES_1	Read-only	JEP106 identification code, bits[6:4]. Together, with PIDR4.DES_2 and PIDR1.DES_0, they indicate the designer of the component and not the implementer, except where the two are the same.

10.13.4.10 css600_tsgen_apb4_slave_1 Peripheral Identification Register 3, PIDR3

The PIDR3 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FEC`

Type

Read-only

Reset

`0x00000000`

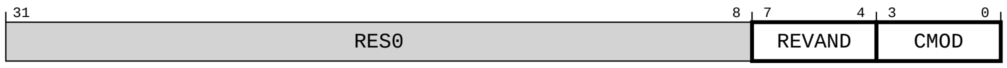
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-352: css600_tsgen_apb4_slave_1_PIDR3



The following table shows the bit assignments.

Table 10-365: PIDR3 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0000`	REVAND	Read-only	This field indicates minor errata fixes specific to this design, for example metal fixes after implementation. In most cases this field is `0x0`.
[3:0]	`0b0000`	CMOD	Read-only	Customer Modified. Where the component is reusable IP, this value indicates if the customer has modified the behavior of the component. In most cases this field is `0x0`.

10.13.4.11 css600_tsgen_apb4_slave_1 Component Identification Register 0, CIDR0

The CIDR0 register is part of the set of component identification registers.

Attributes

Offset

`0x0FF0`

Type

Read-only

Reset

`0x0000000D`

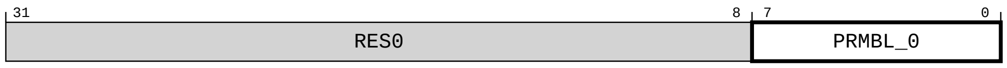
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-353: css600_tsgen_apb4_slave_1_CIDR0



The following table shows the bit assignments.

Table 10-366: CIDR0 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0xD`	PRMBL_0	Read-only	Preamble. Returns `0x0D`.

10.13.4.12 css600_tsgen_apb4_slave_1 Component Identification Register 1, CIDR1

The CIDR1 register is part of the set of component identification registers.

Attributes

Offset

`0x0FF4`

Type

Read-only

Reset

`0x000000F0`

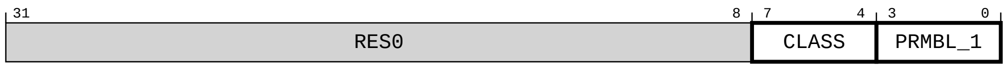
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-354: css600_tsgen_apb4_slave_1_CIDR1



The following table shows the bit assignments.

Table 10-367: CIDR1 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b1111`	CLASS	Read-only	Component class, returns 0xf, indicating CoreLink, PrimeCell, or system component
[3:0]	`0b0000`	PRMBL_1	Read-only	Preamble, returns `0x0`

10.13.4.13 css600_tsgen_apb4_slave_1 Component Identification Register 2, CIDR2

The CIDR2 register is part of the set of component identification registers.

Attributes

Offset

`0x0FF8`

Type

Read-only

Reset

`0x00000005`

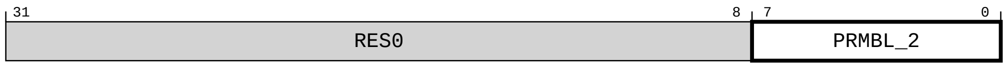
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-355: css600_tsgen_apb4_slave_1_CIDR2



The following table shows the bit assignments.

Table 10-368: CIDR2 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x5`	PRMBL_2	Read-only	Preamble. Returns `0x05`.

10.13.4.14 css600_tsgen_apb4_slave_1 Component Identification Register 3, CIDR3

The CIDR3 register is part of the set of component identification registers.

Attributes

Offset

`0x0FFC`

Type

Read-only

Reset

`0x000000B1`

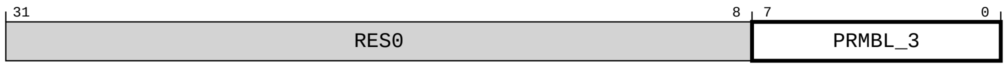
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-356: css600_tsgen_apb4_slave_1_CIDR3



The following table shows the bit assignments.

Table 10-369: CIDR3 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0xB1`	PRMBL_3	Read-only	Preamble. Returns `0xB1`.

10.14 `css600_cti` introduction

This section describes the programmers model of the `css600_cti`.

10.14.1 css600_cti register summary

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

Note

A reset value containing one or more '-' means that this register contains **UNKNOWN** or **IMPLEMENTATION-DEFINED** values. See the relevant register description for more information.

Locations that are not listed in the table are Reserved.

Table 10-370: css600_cti register summary

Offset	Name	Type	Reset	Width	Description
0x0000	CTICONTROL	RW	0x00000000	32	css600_cti CTI Control register, CTICONTROL
0x0010	CTIINTACK	WO	0x00000000	32	css600_cti CTI Interrupt Acknowledge register, CTIINTACK
0x0014	CTIAPPSET	RW	0x00000000	32	css600_cti CTI Application Channel Set register, CTIAPPSET
0x0018	CTIAPPCLEAR	WO	0x00000000	32	css600_cti CTI Application Channel Clear register, CTIAPPCLEAR
0x001C	CTIAPPPULSE	WO	0x00000000	32	css600_cti CTI Application Channel Pulse register, CTIAPPPULSE
0x0020	CTIINEN0	RW	0x00000000	32	css600_cti CTI Trigger 0 to Channel Enable register, CTIINEN0
0x0024	CTIINEN1	RW	0x00000000	32	css600_cti CTI Trigger 1 to Channel Enable register, CTIINEN1
0x0028	CTIINEN2	RW	0x00000000	32	css600_cti CTI Trigger 2 to Channel Enable register, CTIINEN2
…	…	…	…	…	…
0x009C	CTIINEN31	RW	0x00000000	32	css600_cti CTI Trigger 31 to Channel Enable register, CTIINEN31
0x00A0	CTIOUTEN0	RW	0x00000000	32	css600_cti CTI Channel to Trigger 0 Enable register, CTIOUTEN0
0x00A4	CTIOUTEN1	RW	0x00000000	32	css600_cti CTI Channel to Trigger 1 Enable register, CTIOUTEN1
0x00A8	CTIOUTEN2	RW	0x00000000	32	css600_cti CTI Channel to Trigger 2 Enable register, CTIOUTEN2
…	…	…	…	…	…
0x011C	CTIOUTEN31	RW	0x00000000	32	css600_cti CTI Channel to Trigger 31 Enable register, CTIOUTEN31
0x0130	CTITRIGINSTATUS	RO	0x--------	32	css600_cti CTI Trigger Input Status register, CTITRIGINSTATUS
0x0134	CTITRIGOUTSTATUS	RO	0x--------	32	css600_cti CTI Trigger Output Status register, CTITRIGOUTSTATUS
0x0138	CTICHINSTATUS	RO	0x0000000-	32	css600_cti CTI Channel Input Status register, CTICHINSTATUS
0x013C	CTICHOUTSTATUS	RO	0x0000000-	32	css600_cti CTI Channel Output Status register, CTICHOUTSTATUS
0x0140	CTIGATE	RW	0x0000000F	32	css600_cti Enable CTI Channel Gate register, CTIGATE
0x0144	ASICCTRL	RW	0x00000000	32	css600_cti External Multiplexer Control register, ASICCTRL
0x0EE4	ITCHOUT	RW	0x00000000	32	css600_cti Integration Test Channel Output register, ITCHOUT
0x0EE8	ITTRIGOUT	RW	0x00000000	32	css600_cti Integration Test Trigger Output register, ITTRIGOUT
0x0EF4	ITCHIN	RO	0x00000000	32	css600_cti Integration Test Channel Input register, ITCHIN
0x0EF8	ITTRIGIN	RO	0x00000000	32	css600_cti Integration Test Trigger Input register, ITTRIGIN
0x0F00	ITCTRL	RW	0x00000000	32	css600_cti Integration Mode Control Register, ITCTRL
0x0FA0	CLAIMSET	RW	0x0000000F	32	css600_cti Claim Tag Set Register, CLAIMSET
0x0FA4	CLAIMCLR	RW	0x00000000	32	css600_cti Claim Tag Clear Register, CLAIMCLR
0x0FA8	DEVAFF0	RO	0x--------	32	css600_cti Device Affinity register 0, DEVAFF0
0x0FAC	DEVAFF1	RO	0x--------	32	css600_cti Device Affinity register 1, DEVAFF1

Offset	Name	Type	Reset	Width	Description
`0x0FB8`	AUTHSTATUS	RO	`0x0000000-`	32	css600_cti Authentication Status Register, AUTHSTATUS
`0x0FBC`	DEVARCH	RO	`0x47701A14`	32	css600_cti Device Architecture Register, DEVARCH
`0x0FC8`	DEVID	RO	`0x0104----`	32	css600_cti Device Configuration Register, DEVID
`0x0FCC`	DEVTYPE	RO	`0x00000014`	32	css600_cti Device Type Identifier Register, DEVTYPE
`0x0FD0`	PIDR4	RO	`0x00000004`	32	css600_cti Peripheral Identification Register 4, PIDR4
`0x0FD4`	PIDR5	RO	`0x00000000`	32	css600_cti Peripheral Identification Register 5, PIDR5
`0x0FD8`	PIDR6	RO	`0x00000000`	32	css600_cti Peripheral Identification Register 6, PIDR6
`0x0FDC`	PIDR7	RO	`0x00000000`	32	css600_cti Peripheral Identification Register 7, PIDR7
`0x0FE0`	PIDR0	RO	`0x000000ED`	32	css600_cti Peripheral Identification Register 0, PIDR0
`0x0FE4`	PIDR1	RO	`0x000000B9`	32	css600_cti Peripheral Identification Register 1, PIDR1
`0x0FE8`	PIDR2	RO	`0x0000003B`	32	css600_cti Peripheral Identification Register 2, PIDR2
`0x0FEC`	PIDR3	RO	`0x00000000`	32	css600_cti Peripheral Identification Register 3, PIDR3
`0x0FF0`	CIDR0	RO	`0x0000000D`	32	css600_cti Component Identification Register 0, CIDR0
`0x0FF4`	CIDR1	RO	`0x00000090`	32	css600_cti Component Identification Register 1, CIDR1
`0x0FF8`	CIDR2	RO	`0x00000005`	32	css600_cti Component Identification Register 2, CIDR2
`0x0FFC`	CIDR3	RO	`0x000000B1`	32	css600_cti Component Identification Register 3, CIDR3

10.14.2 Register descriptions

This section describes the `css600_cti` registers.

css600_cti register summary provides cross references to individual registers.

Note

The number of bits implemented in the CTIINTACK, CTITRIGINSTATUS, CTITRIGOUTSTATUS, ITTRIGOUT, and ITTRIGIN registers depends on the number of triggers <n> implemented. There is 1 bit in each register for each implemented trigger, bits[<n>-1:0], with bits[31:<n>] Reserved. Registers CTIINEN0..CTIINEN31 and CTIOUTEN0..CTIOUTEN31 are all implemented, regardless of the number of triggers implemented. Reads from registers that are associated with unimplemented triggers return **UNDEFINED** values, and writes have no effect.

10.14.2.1 css600_cti CTI Control register, CTICONTROL

The CTICONTROL register enables and disables the CTI.

Attributes

Offset

`0x0000`

Type

Read-write

Reset

`0x00000000`

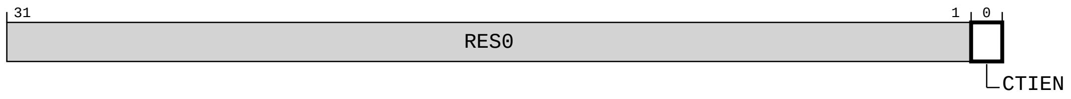
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-357: css600_cti_CTICONTROL



The following table shows the bit assignments.

Table 10-371: CTICONTROL attributes

Bits	Reset value	Name	Type	Function
[31:1]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[0]	`0b0`	CTIEN	Read-write	Enable control: **0** CTI disabled **1** CTI enabled

10.14.2.2 css600_cti CTI Interrupt Acknowledge register, CTIINTACK

The CTIINTACK register acknowledges trigger outputs. It is a bit map that allows selective clearing of trigger output events. If the SW_HANDSHAKE parameter for a trigger output is set, indicating that the output latches HIGH when an event is sent to that output, then the output remains HIGH until the corresponding INTACK bit is written to a 1. A write of a bit to 0 has no effect. This allows different software threads to be responsible for clearing different trigger outputs without needing to perform a read-modify-write operation to find which bits are set.

Attributes

Offset

`0x0010`

Type

Write-only

Reset

`0x00000000`

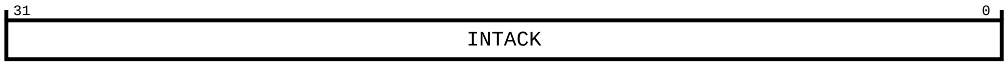
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-358: css600_cti_CTIINTACK



The following table shows the bit assignments.

Table 10-372: CTIINTACK attributes

Bits	Reset value	Name	Type	Function
[31:0]	`0x0`	INTACK	Write-only	Acknowledges the corresponding event_out output

10.14.2.3 css600_cti CTI Application Channel Set register, CTIAPPSET

The CTIAPPSET register allows software to set any channel output. Software can use this register to generate a channel event in place of a hardware source on a trigger input. This register must not be used in a system where all events are sent as single-cycle pulses. It is only retained for compatibility with older systems and software. The register is implemented before the CTIGATE register. Therefore, for the channel event to propagate outside the CTI, the corresponding CTIGATE bit must be set to 1.

Attributes

Offset

`0x0014`

Type

Read-write

Reset

`0x00000000`

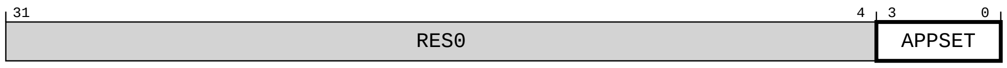
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-359: css600_cti_CTIAPPSET



The following table shows the bit assignments.

Table 10-373: CTIAPPSET attributes

Bits	Reset value	Name	Type	Function
[31:4]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[3:0]	`0b0000`	APPSET	Read-write	Sets the corresponding internal channel flag: **0** Read: application channel is inactive. Write: has no effect. **1** Read: application channel is active. Write: sets the channel output.

10.14.2.4 css600_cti CTI Application Channel Clear register, CTIAPPCLEAR

The CTIAPPCLEAR register allows software to clear any channel output. Software can use this register to clear a channel event in place of a hardware source on a trigger input. This register must not be used in a system where all events are sent as single-cycle pulses. It is only retained for compatibility with older systems and software. The register is implemented before the CTIGATE register. Therefore, for the channel event to propagate outside the CTI, the corresponding CTIGATE bit must be set to 1.

Attributes

Offset

`0x0018`

Type

Write-only

Reset

`0x00000000`

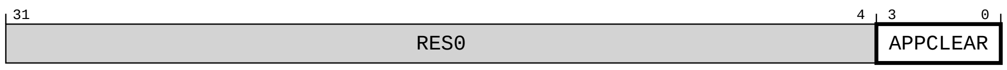
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-360: css600_cti_CTIAPPCLEAR



The following table shows the bit assignments.

Table 10-374: CTIAPPCLEAR attributes

Bits	Reset value	Name	Type	Function
[31:4]	`0x0`	RES0	Write-only	Reserved bit or field with SBZP behavior

Bits	Reset value	Name	Type	Function
[3:0]	`0b0000`	APPCLEAR	Write-only	Clears the corresponding internal channel flag. **0** No effect. **1** Clears the channel output.

10.14.2.5 css600_cti CTI Application Channel Pulse register, CTIAPPPULSE

The application channel pulse register allows software to pulse any channel output. This register can be used by software to pulse a channel event in place of a hardware source on a trigger input. The register is implemented before the CTIGATE register so, for the channel event to propagate outside the CTI, it is necessary for the corresponding CTIGATE bit to be 1.

Attributes

Offset

`0x001C`

Type

Write-only

Reset

`0x00000000`

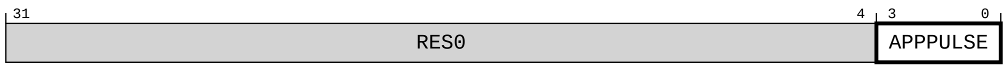
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-361: css600_cti_CTIAPPPULSE



The following table shows the bit assignments.

Table 10-375: CTIAPPPULSE attributes

Bits	Reset value	Name	Type	Function
[31:4]	`0x0`	RES0	Write-only	Reserved bit or field with SBZP behavior
[3:0]	`0b0000`	APPPULSE	Write-only	Pulses the channel outputs. **0** No effect. **1** Pulse channel event for one clk cycle.

10.14.2.6 css600_cti CTI Trigger 0 to Channel Enable register, CTIINEN0

This register maps trigger inputs to channels in the cross trigger system. The CTIINEN registers are bit maps that allow the trigger input to be mapped to any channel output, including none (`0x0`) and all (0xF). There is one register per trigger input, so it is possible to map any combination of trigger inputs to any channel outputs.

Attributes

Offset

`0x0020`

Type

Read-write

Reset

`0x00000000`

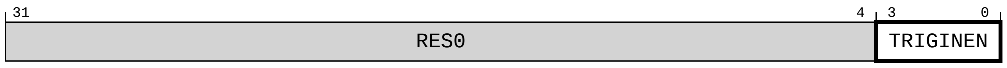
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-362: css600_cti_CTIINEN0



The following table shows the bit assignments.

Table 10-376: CTIINEN0 attributes

Bits	Reset value	Name	Type	Function
[31:4]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[3:0]	`0b0000`	TRIGINEN	Read-write	Trigger input to channel mapping. **0** Input trigger 0 events are ignored by the corresponding channel. **1** When an event is received on event_in[0], generate an event on the channel corresponding to this bit.

10.14.2.7 css600_cti CTI Trigger 1 to Channel Enable register, CTIINEN1

This register maps trigger inputs to channels in the cross trigger system. The CTIINEN registers are bit maps that allow the trigger input to be mapped to any channel output, including none (`0x0`) and

all (0xF). There is one register per trigger input, so it is possible to map any combination of trigger inputs to any channel outputs.

CTIINEN1-31 registers are **IMPLEMENTATION-DEFINED**. If they are not implemented, the locations are RO and return 0. If they are implemented, they are RW and return a reset value of 0.

Attributes

Offset

`0x0024`

Type

Read-write

Reset

`0x00000000`

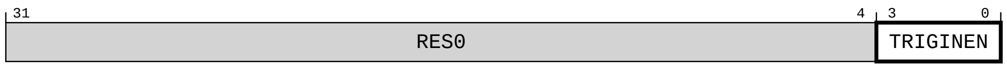
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-363: css600_cti_CTIINEN1



The following table shows the bit assignments.

Table 10-377: CTIINEN1 attributes

Bits	Reset value	Name	Type	Function
[31:4]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[3:0]	`0b0000`	TRIGINEN	Read-write	Trigger input to channel mapping. **0** Input trigger 1 events are ignored by the corresponding channel. **1** When an event is received on event_in[1], generate an event on the channel corresponding to this bit.

10.14.2.8 css600_cti CTI Trigger 2 to Channel Enable register, CTIINEN2

This register maps trigger inputs to channels in the cross trigger system. The CTIINEN registers are bit maps that allow the trigger input to be mapped to any channel output, including none (`0x0`) and

all (0xF). There is one register per trigger input, so it is possible to map any combination of trigger inputs to any channel outputs.

CTIINEN1-31 registers are **IMPLEMENTATION-DEFINED**. If they are not implemented, the locations are RO and return 0. If they are implemented, they are RW and return a reset value of 0.

Attributes

Offset

`0x0028`

Type

Read-write

Reset

`0x00000000`

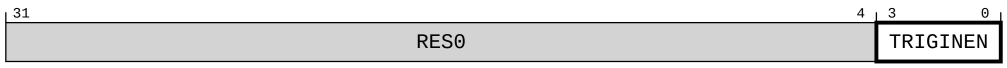
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-364: css600_cti_CTIINEN2



The following table shows the bit assignments.

Table 10-378: CTIINEN2 attributes

Bits	Reset value	Name	Type	Function
[31:4]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[3:0]	`0b0000`	TRIGINEN	Read-write	Trigger input to channel mapping. **0** Input trigger 2 events are ignored by the corresponding channel. **1** When an event is received on event_in[2], generate an event on the channel corresponding to this bit.

10.14.2.9 css600_cti CTI Trigger 31 to Channel Enable register, CTIINEN31

This register maps trigger inputs to channels in the cross trigger system. The CTIINEN registers are bit maps that allow the trigger input to be mapped to any channel output, including none (`0x0`) and

all (0xF). There is one register per trigger input, so it is possible to map any combination of trigger inputs to any channel outputs.

CTIINEN1-31 registers are **IMPLEMENTATION-DEFINED**. If they are not implemented, the locations are RO and return 0. If they are implemented, they are RW and return a reset value of 0.

Attributes

Offset

`0x009C`

Type

Read-write

Reset

`0x00000000`

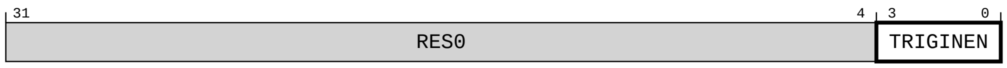
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-365: css600_cti_CTIINEN31



The following table shows the bit assignments.

Table 10-379: CTIINEN31 attributes

Bits	Reset value	Name	Type	Function
[31:4]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[3:0]	`0b0000`	TRIGINEN	Read-write	Trigger input to channel mapping. **0** Input trigger 31 events are ignored by the corresponding channel. **1** When an event is received on event_in[31], generate an event on the channel corresponding to this bit.

10.14.2.10 css600_cti CTI Channel to Trigger 0 Enable register, CTIOUTEN0

This register maps channels in the cross trigger system to trigger outputs. The CTIOUTEN registers are bit maps that allow any channel input to be mapped to the trigger output, including none (`0x0`) and all (0xF). There is one register per trigger output so it is possible to map any channel input to any combination of trigger outputs.

Attributes

Offset

`0x00A0`

Type

Read-write

Reset

`0x00000000`

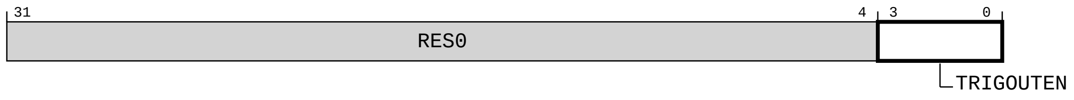
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-366: css600_cti_CTIOUTEN0



The following table shows the bit assignments.

Table 10-380: CTIOUTEN0 attributes

Bits	Reset value	Name	Type	Function
[31:4]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[3:0]	`0b0000`	TRIGOUTEN	Read-write	Channel to trigger output mapping. **0** The corresponding channel is ignored by the output trigger0. **1** When an event occurs on the channel corresponding to this bit, generate an event on event_out[0].

10.14.2.11 css600_cti CTI Channel to Trigger 1 Enable register, CTIOUTEN1

This register maps channels in the cross trigger system to trigger outputs. The CTIOUTEN registers are bit maps that allow any channel input to be mapped to the trigger output, including none (`0x0`) and all (0xF). There is one register per trigger output so it is possible to map any channel input to any combination of trigger outputs.

CTIOUTEN1-31 registers are **IMPLEMENTATION-DEFINED**. If they are not implemented, the locations are RO and return 0. If they are implemented, they are RW and return a reset value of 0.

Attributes

Offset

`0x00A4`

Type

Read-write

Reset

`0x00000000`

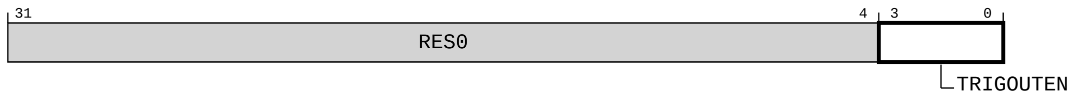
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-367: css600_cti_CTIOUTEN1



The following table shows the bit assignments.

Table 10-381: CTIOUTEN1 attributes

Bits	Reset value	Name	Type	Function
[31:4]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[3:0]	`0b0000`	TRIGOUTEN	Read-write	Channel to trigger output mapping. **0** The corresponding channel is ignored by the output trigger1. **1** When an event occurs on the channel corresponding to this bit, generate an event on event_out[1].

10.14.2.12 css600_cti CTI Channel to Trigger 2 Enable register, CTIOUTEN2

This register maps channels in the cross trigger system to trigger outputs. The CTIOUTEN registers are bit maps that allow any channel input to be mapped to the trigger output, including none (`0x0`) and all (0xF). There is one register per trigger output so it is possible to map any channel input to any combination of trigger outputs.

CTIOUTEN1-31 registers are **IMPLEMENTATION-DEFINED**. If they are not implemented, the locations are RO and return 0. If they are implemented, they are RW and return a reset value of 0.

Attributes

Offset

`0x00A8`

Type

Read-write

Reset

`0x00000000`

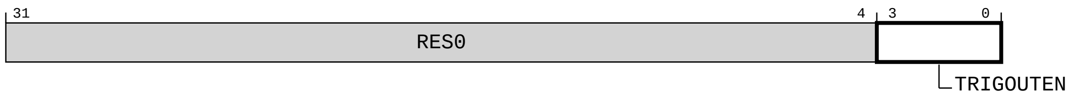
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-368: css600_cti_CTIOUTEN2



The following table shows the bit assignments.

Table 10-382: CTIOUTEN2 attributes

Bits	Reset value	Name	Type	Function
[31:4]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[3:0]	`0b0000`	TRIGOUTEN	Read-write	Channel to trigger output mapping. **0** The corresponding channel is ignored by the output trigger2. **1** When an event occurs on the channel corresponding to this bit, generate an event on event_out[2].

10.14.2.13 css600_cti CTI Channel to Trigger 31 Enable register, CTIOUTEN31

This register maps channels in the cross trigger system to trigger outputs. The CTIOUTEN registers are bit maps that allow any channel input to be mapped to the trigger output, including none (`0x0`) and all (0xF). There is one register per trigger output so it is possible to map any channel input to any combination of trigger outputs.

CTIOUTEN1-31 registers are **IMPLEMENTATION-DEFINED**. If they are not implemented, the locations are RO and return 0. If they are implemented, they are RW and return a reset value of 0.

Attributes

Offset

`0x011C`

Type

Read-write

Reset

`0x00000000`

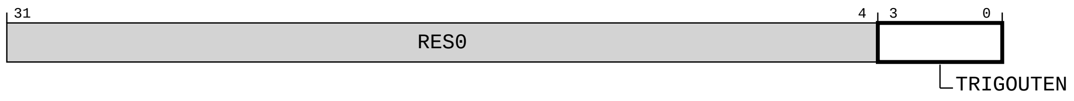
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-369: css600_cti_CTIOUTEN31



The following table shows the bit assignments.

Table 10-383: CTIOUTEN31 attributes

Bits	Reset value	Name	Type	Function
[31:4]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[3:0]	`0b0000`	TRIGOUTEN	Read-write	Channel to trigger output mapping. **0** The corresponding channel is ignored by the output trigger31. **1** When an event occurs on the channel corresponding to this bit, generate an event on event_out[31].

10.14.2.14 css600_cti CTI Trigger Input Status register, CTITRIGINSTATUS

Trigger input status. If the event_in input is driven by a source that generates single cycle pulses, this register is generally read as 0.

Attributes

Offset

`0x0130`

Type

Read-only

Reset

`0x--------`

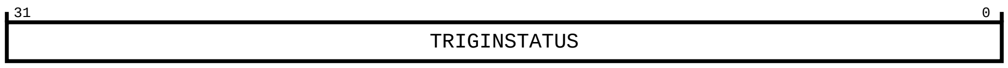
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-370: css600_cti_CTITRIGINSTATUS



The following table shows the bit assignments.

Table 10-384: CTITRIGINSTATUS attributes

Bits	Reset value	Name	Type	Function
[31:0]	UNKNOWN	TRIGINSTATUS	Read-only	Trigger input status. **0** One bit per trigger input. 0 means that the input is LOW. **1** One bit per trigger input. 1 means that the input is HIGH.

10.14.2.15 css600_cti CTI Trigger Output Status register, CTITRIGOUTSTATUS

Trigger output status. The register only has meaning if the trigger source drives static levels, rather than pulses.

Attributes

Offset

`0x0134`

Type

Read-only

Reset

`0x--------`

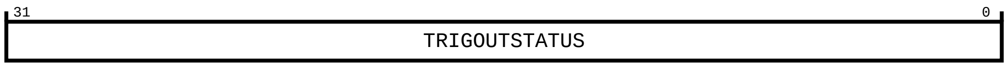
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-371: css600_cti_CTITRIGOUTSTATUS



The following table shows the bit assignments.

Table 10-385: CTITRIGOUTSTATUS attributes

Bits	Reset value	Name	Type	Function
[31:0]	UNKNOWN	TRIGOUTSTATUS	Read-only	Trigger output status. **0** One bit per trigger output. 0 means that the output is LOW. **1** One bit per trigger output. 1 means that the output is HIGH.

10.14.2.16 css600_cti CTI Channel Input Status register, CTICHINSTATUS

Channel input status. If the channel input is driven by a source that generates single cycle pulses, this register is generally read as 0.

Attributes

Offset

`0x0138`

Type

Read-only

Reset

`0x0000000-`

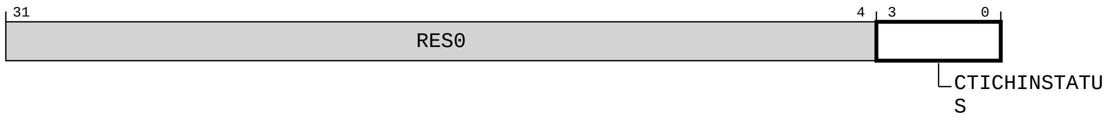
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-372: css600_cti_CTICHINSTATUS



The following table shows the bit assignments.

Table 10-386: CTICHINSTATUS attributes

Bits	Reset value	Name	Type	Function
[31:4]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[3:0]	UNKNOWN	CTICHINSTATUS	Read-only	Channel input status. **0** One bit per channel input. 0 means that the input is LOW. **1** One bit per channel input. 1 means that the input is HIGH.

10.14.2.17 css600_cti CTI Channel Output Status register, CTICHOUTSTATUS

Channel output status. The register only has meaning if the trigger source drives static levels, rather than pulses.

Attributes

Offset

`0x013C`

Type

Read-only

Reset

`0x0000000-`

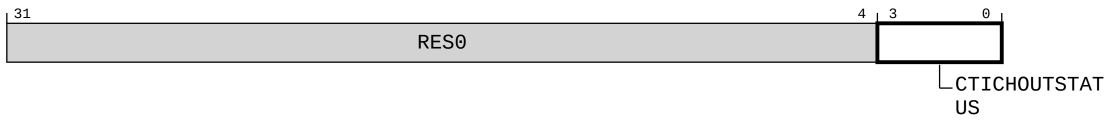
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-373: css600_cti_CTICHOUTSTATUS



The following table shows the bit assignments.

Table 10-387: CTICHOUTSTATUS attributes

Bits	Reset value	Name	Type	Function
[31:4]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[3:0]	UNKNOWN	CTICHOUTSTATUS	Read-only	Channel output status. **0** One bit per channel output. 0 means that the output is LOW. **1** One bit per channel output. 1 means that the output is HIGH.

10.14.2.18 css600_cti Enable CTI Channel Gate register, CTIGATE

Channel output gate.

Attributes

Offset

`0x0140`

Type

Read-write

Reset

`0x0000000F`

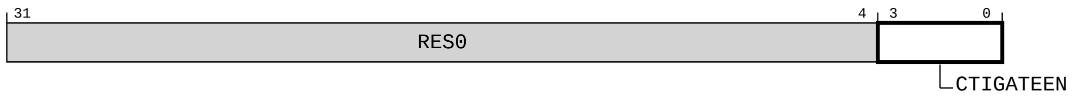
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-374: css600_cti_CTIGATE



The following table shows the bit assignments.

Table 10-388: CTIGATE attributes

Bits	Reset value	Name	Type	Function
[31:4]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[3:0]	`0b1111`	CTIGATEEN	Read-write	Enables the propagation of channel events out of the CTI, one bit per channel. **0** Disable a channel from propagating. **1** Enable channel propagation.

10.14.2.19 css600_cti External Multiplexer Control register, ASICCTRL

I/O port control.

Attributes

Offset

`0x0144`

Type

Read-write

Reset

`0x00000000`

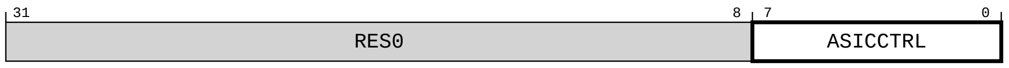
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-375: css600_cti_ASICCTRL



The following table shows the bit assignments.

Table 10-389: ASICCTRL attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	ASICCTRL	Read-write	Set and clear external output signal. **0** Clear output bit to 0. **1** Set output bit to 1.

10.14.2.20 css600_cti Integration Test Channel Output register, ITCHOUT

Integration test mode register, used to generate channel events. Writing to the register creates a single pulse on the output. ITCHOUT is self-clearing.

Attributes

Offset

`0x0EE4`

Type

Read-write

Reset

`0x00000000`

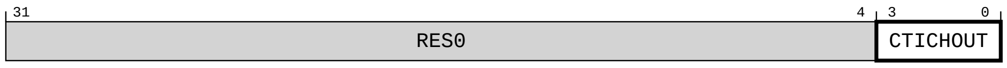
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-376: css600_cti_ITCHOUT



The following table shows the bit assignments.

Table 10-390: ITCHOUT attributes

Bits	Reset value	Name	Type	Function
[31:4]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[3:0]	`0b0000`	CTICHOUT	Write-only	Pulses the channel outputs. **0** No effect. **1** Pulse channel event for one clk cycle.

10.14.2.21 css600_cti Integration Test Trigger Output register, ITTRIGOUT

Integration test mode register, used to generate trigger events.

Attributes

Offset

`0x0EE8`

Type

Read-write

Reset

`0x00000000`

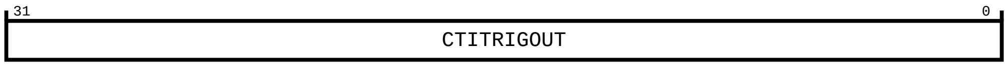
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-377: css600_cti_ITTRIGOUT



The following table shows the bit assignments.

Table 10-391: ITTRIGOUT attributes

Bits	Reset value	Name	Type	Function
[31:0]	`0x0`	CTITRIGOUT	Read-write	Set/clear trigger output signal. Reads return the value in the register if SW_HANDSHAKE=1, otherwise 0 is returned if SW_HANDSHAKE=0. Writes: **0** Clears the trigger output if SW_HANDSHAKE=1, no effect if SW_HANDSHAKE=0. **1** Sets the trigger output if SW_HANDSHAKE=1, pulses trigger output if SW_HANDSHAKE=0.

10.14.2.22 css600_cti Integration Test Channel Input register, ITCHIN

Integration test mode register, used to view channel events. The integration test register includes a latch that is set when a pulse is received on a channel input. When read, a register bit reads as 1 if the channel has received a pulse since it was last read. The act of reading the register automatically clears the 1 to a 0. When performing integration testing it is therefore important to coordinate the setting of event latches and reading/clearing them.

Attributes

Offset

`0x0EF4`

Type

Read-only

Reset

`0x00000000`

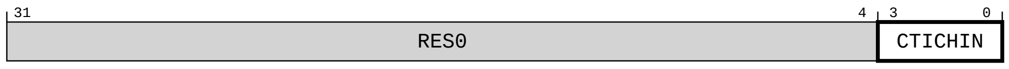
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-378: css600_cti_ITCHIN



The following table shows the bit assignments.

Table 10-392: ITCHIN attributes

Bits	Reset value	Name	Type	Function
[31:4]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[3:0]	`0b0000`	CTICHIN	Read-only	Reads the latched value of the channel inputs.

10.14.2.23 css600_cti Integration Test Trigger Input register, ITTRIGIN

Integration test mode register, used to view trigger events. The integration test register includes a latch that is set when a pulse is received on a trigger input. When read, a register bit reads as 1 if the trigger input has received a pulse since it was last read. The act of reading the register automatically clears the 1 to a 0. When performing integration testing it is therefore important to coordinate the setting of event latches and reading/clearing them.

Attributes

Offset

`0x0EF8`

Type

Read-only

Reset

`0x00000000`

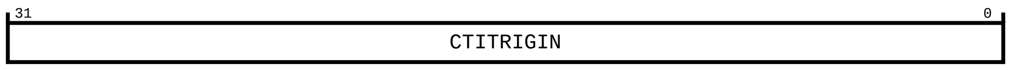
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-379: css600_cti_ITTRIGIN



The following table shows the bit assignments.

Table 10-393: ITTRIGIN attributes

Bits	Reset value	Name	Type	Function
[31:0]	`0x0`	CTITRIGIN	Read-only	Reads the latched value of the trigger inputs.

10.14.2.24 css600_cti Integration Mode Control Register, ITCTRL

The Integration Mode Control register is used to enable topology detection.

Attributes

Offset

`0x0F00`

Type

Read-write

Reset

`0x00000000`

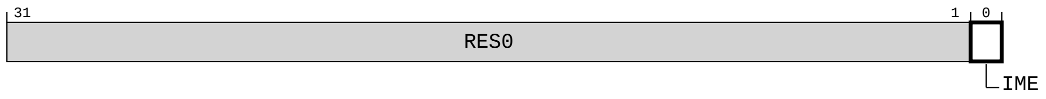
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-380: css600_cti_ITCTRL



The following table shows the bit assignments.

Table 10-394: ITCTRL attributes

Bits	Reset value	Name	Type	Function
[31:1]	`0x0`	RES0	Read-write	Reserved bit or field with SBZP behavior
[0]	`0b0`	IME	Read-write	Integration Mode Enable. When set, the component enters integration mode, enabling topology detection or integration testing to be performed.

10.14.2.25 css600_cti Claim Tag Set Register, CLAIMSET

This register forms one half of the claim tag value. On writes, this location enables individual bits to be set. On reads, it returns the number of bits that can be set.

Attributes

Offset

`0x0FA0`

Type

Read-write

Reset

`0x0000000F`

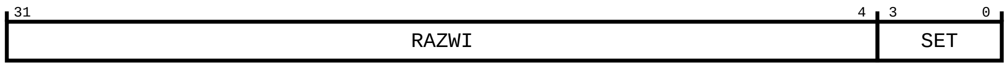
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-381: css600_cti_CLAIMSET



The following table shows the bit assignments.

Table 10-395: CLAIMSET attributes

Bits	Reset value	Name	Type	Function
[31:4]	`0x0`	RAZ/WI	Read-write	RAZ/WI
[3:0]	`0b1111`	SET	Read-write	A bit-programmable register bank that sets the claim tag value. A read returns a logic 1 for all implemented locations.

10.14.2.26 css600_cti Claim Tag Clear Register, CLAIMCLR

This register forms one half of the claim tag value. On writes, this location enables individual bits to be cleared. On reads, it returns the current claim tag value.

Attributes

Offset

`0x0FA4`

Type

Read-write

Reset

`0x00000000`

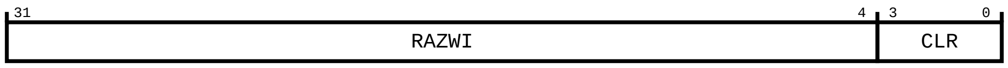
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-382: css600_cti_CLAIMCLR



The following table shows the bit assignments.

Table 10-396: CLAIMCLR attributes

Bits	Reset value	Name	Type	Function
[31:4]	`0x0`	RAZ/WI	Read-write	RAZ/WI
[3:0]	`0b0000`	CLR	Read-write	A bit-programmable register bank that clears the claim tag value. It is zero at reset. It is used by software agents to signal to each other ownership of the hardware. It has no direct effect on the hardware itself.

10.14.2.27 css600_cti Device Affinity register 0, DEVAFF0

Enables a debugger to determine if two components have an affinity with each other.

Attributes

Offset

`0x0FA8`

Type

Read-only

Reset

`0x--------`

Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-383: css600_cti_DEVAFF0

31
0
DEVAFF0

The following table shows the bit assignments.

Table 10-397: DEVAFF0 attributes

Bits	Reset value	Name	Type	Function
[31:0]	IMPLEMENTATION DEFINED	DEVAFF0	Read-only	Lower 32-bits of DEVAFF. The value is set by the devaff[31:0] tie-off inputs.

10.14.2.28 css600_cti Device Affinity register 1, DEVAFF1

Enables a debugger to determine if two components have an affinity with each other.

Attributes

Offset

`0x0FAC`

Type

Read-only

Reset

`0x--------`

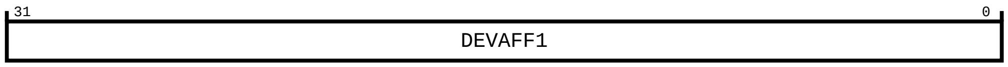
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-384: css600_cti_DEVAFF1



The following table shows the bit assignments.

Table 10-398: DEVAFF1 attributes

Bits	Reset value	Name	Type	Function
[31:0]	IMPLEMENTATION DEFINED	DEVAFF1	Read-only	Upper 32-bits of DEVAFF. The value is set by the devaff[63:32] tie-off inputs.

10.14.2.29 css600_cti Authentication Status Register, AUTHSTATUS

Reports the current status of the authentication control signals.

Attributes

Offset

`0x0FB8`

Type

Read-only

Reset

`0x0000000-`

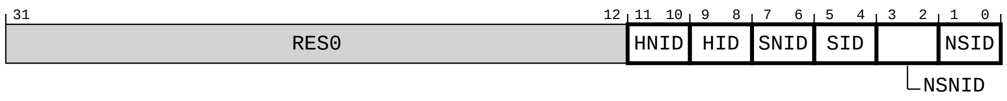
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-385: css600_cti_AUTHSTATUS



The following table shows the bit assignments.

Table 10-399: AUTHSTATUS attributes

Bits	Reset value	Name	Type	Function
[31:12]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[11:10]	`0b00`	HNID	Read-only	Hypervisor non-invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.
[9:8]	`0b00`	HID	Read-only	Hypervisor invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.
[7:6]	`0b00`	SNID	Read-only	Secure non-invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.
[5:4]	`0b00`	SID	Read-only	Secure invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.
[3:2]	UNKNOWN	NSNID	Read-only	Non-secure non-invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.
[1:0]	UNKNOWN	NSID	Read-only	Non-secure invasive debug. **`0x0`** Functionality not implemented or controlled elsewhere. **`0x1`** Reserved. **`0x2`** Functionality disabled. **`0x3`** Functionality enabled.

10.14.2.30 css600_cti Device Architecture Register, DEVARCH

Identifies the architect and architecture of a CoreSight component. The architect might differ from the designer of a component, for example Arm defines the architecture but another company designs and implements the component.

Attributes

Offset

`0x0FBC`

Type

Read-only

Reset

`0x47701A14`

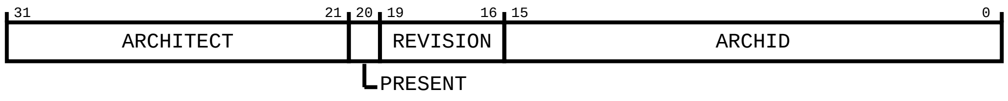
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-386: css600_cti_DEVARCH



The following table shows the bit assignments.

Table 10-400: DEVARCH attributes

Bits	Reset value	Name	Type	Function
[31:21]	`0x23B`	ARCHITECT	Read-only	Returns 0x23B, denoting Arm as architect of the component
[20]	`0b1`	PRESENT	Read-only	Returns 1, indicating that the DEVARCH register is present
[19:16]	`0b0000`	REVISION	Read-only	Architecture revision. Returns the revision of the architecture that the ARCHID field specifies.
[15:0]	`0x1A14`	ARCHID	Read-only	Architecture ID. Returns 0x1A14, identifying Cross Trigger Interface architecture v2.

10.14.2.31 css600_cti Device Configuration Register, DEVID

This register is **IMPLEMENTATION DEFINED** for each Part Number and Designer. The register indicates the capabilities of the component.

Attributes

Offset

`0x0FC8`

Type

Read-only

Reset

`0x0104----`

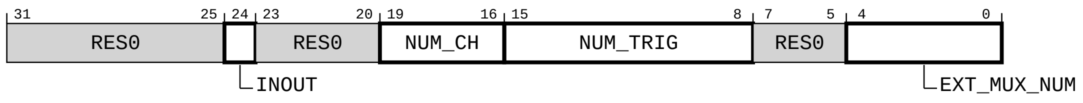
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-387: css600_cti_DEVID



The following table shows the bit assignments.

Table 10-401: DEVID attributes

Bits	Reset value	Name	Type	Function
[31:25]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[24]	`0b1`	INOUT	Read-only	Indicates channel inputs are also masked by the CTIGATE register. Always 1.
[23:20]	`0b0000`	RES0	Read-only	Reserved bit or field with SBZP behavior
[19:16]	`0b0100`	NUM_CH	Read-only	The number of channels. Always 4.
[15:8]	**IMPLEMENTATION DEFINED**	NUM_TRIG	Read-only	Indicates the maximum number of triggers - the maximum of the two parameters, NUM_EVENT_SLAVES and NUM_EVENT_MASTERS.
[7:5]	`0b000`	RES0	Read-only	Reserved bit or field with SBZP behavior
[4:0]	**IMPLEMENTATION DEFINED**	EXT_MUX_NUM	Read-only	Indicates the value of the EXT_MUX_NUM parameter, which determines if there is any external multiplexing on the trigger inputs and outputs. 0 indicates no multiplexing.

10.14.2.32 css600_cti Device Type Identifier Register, DEVTYPE

A debugger can use this register to get information about a component that has an unrecognized Part number.

Attributes

Offset

`0x0FCC`

Type

Read-only

Reset

`0x00000014`

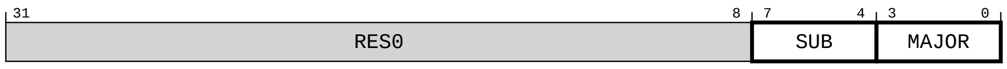
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-388: css600_cti_DEVTYPE



The following table shows the bit assignments.

Table 10-402: DEVTYPE attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0001`	SUB	Read-only	Minor classification. Returns `0x1`, indicating this component is a Trigger-Matrix.
[3:0]	`0b0100`	MAJOR	Read-only	Major classification. Returns `0x4`, indicating this component performs Debug Control.

10.14.2.33 css600_cti Peripheral Identification Register 4, PIDR4

The PIDR4 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FD0`

Type

Read-only

Reset

`0x00000004`

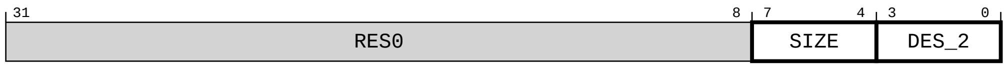
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-389: css600_cti_PIDR4



The following table shows the bit assignments.

Table 10-403: PIDR4 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0000`	SIZE	Read-only	Indicates the memory size that is used by this component. Returns 0 indicating that the component uses an **UNKNOWN** number of 4KB blocks. Using the SIZE field to indicate the size of the component is deprecated.
[3:0]	`0b0100`	DES_2	Read-only	JEP106 continuation code. Together, with PIDR2.DES_1 and PIDR1.DES_0, they indicate the designer of the component and not the implementer, except where the two are the same.

10.14.2.34 css600_cti Peripheral Identification Register 5, PIDR5

The PIDR5 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FD4`

Type

Read-only

Reset

`0x00000000`

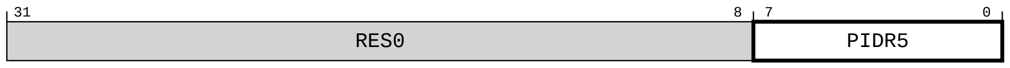
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-390: css600_cti_PIDR5



The following table shows the bit assignments.

Table 10-404: PIDR5 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	PIDR5	Read-only	Reserved.

10.14.2.35 css600_cti Peripheral Identification Register 6, PIDR6

The PIDR6 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FD8`

Type

Read-only

Reset

`0x00000000`

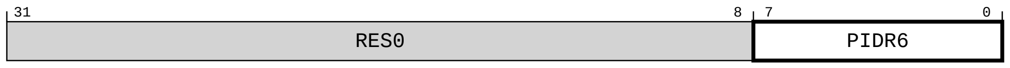
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-391: css600_cti_PIDR6



The following table shows the bit assignments.

Table 10-405: PIDR6 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	PIDR6	Read-only	Reserved.

10.14.2.36 css600_cti Peripheral Identification Register 7, PIDR7

The PIDR7 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FDC`

Type

Read-only

Reset

`0x00000000`

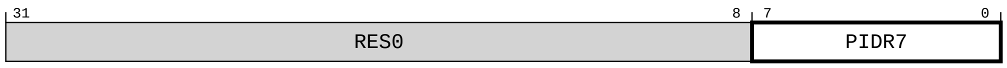
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-392: css600_cti_PIDR7



The following table shows the bit assignments.

Table 10-406: PIDR7 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x0`	PIDR7	Read-only	Reserved.

10.14.2.37 css600_cti Peripheral Identification Register 0, PIDR0

The PIDR0 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FE0`

Type

Read-only

Reset

`0x000000ED`

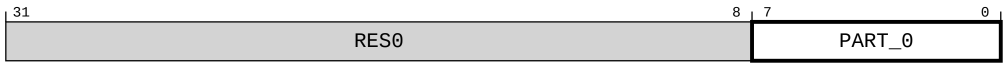
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-393: css600_cti_PIDR0



The following table shows the bit assignments.

Table 10-407: PIDR0 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	0xED	PART_0	Read-only	Part number, bits[7:0]. Taken together with PIDR1.PART_1 it indicates the component. The Part Number is selected by the designer of the component.

10.14.2.38 css600_cti Peripheral Identification Register 1, PIDR1

The PIDR1 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FE4`

Type

Read-only

Reset

`0x000000B9`

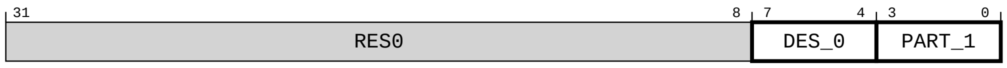
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-394: css600_cti_PIDR1



The following table shows the bit assignments.

Table 10-408: PIDR1 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b1011`	DES_0	Read-only	JEP106 identification code, bits[3:0]. Together, with PIDR4.DES_2 and PIDR2.DES_1, they indicate the designer of the component and not the implementer, except where the two are the same.
[3:0]	`0b1001`	PART_1	Read-only	Part number, bits[11:8]. Taken together with PIDR0.PART_0 it indicates the component. The Part Number is selected by the designer of the component.

10.14.2.39 css600_cti Peripheral Identification Register 2, PIDR2

The PIDR2 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FE8`

Type

Read-only

Reset

`0x0000003B`

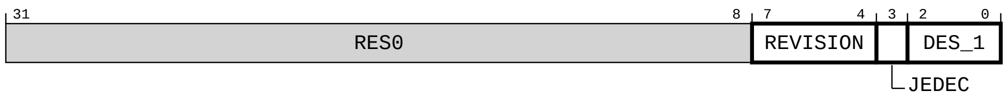
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-395: css600_cti_PIDR2



The following table shows the bit assignments.

Table 10-409: PIDR2 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0011`	REVISION	Read-only	Revision. It is an incremental value starting at `0x0` for the first design of a component. See the Component list in Chapter 1 for information on the RTL revision of the component.

Bits	Reset value	Name	Type	Function
[3]	`0b1`	JEDEC	Read-only	1 - Always set. Indicates that a JEDEC assigned value is used.
[2:0]	`0b011`	DES_1	Read-only	JEP106 identification code, bits[6:4]. Together, with PIDR4.DES_2 and PIDR1.DES_0, they indicate the designer of the component and not the implementer, except where the two are the same.

10.14.2.40 css600_cti Peripheral Identification Register 3, PIDR3

The PIDR3 register is part of the set of peripheral identification registers.

Attributes

Offset

`0x0FEC`

Type

Read-only

Reset

`0x00000000`

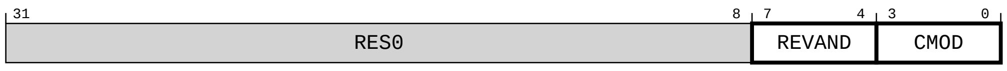
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-396: css600_cti_PIDR3



The following table shows the bit assignments.

Table 10-410: PIDR3 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b0000`	REVAND	Read-only	This field indicates minor errata fixes specific to this design, for example metal fixes after implementation. In most cases this field is `0x0`.
[3:0]	`0b0000`	CMOD	Read-only	Customer Modified. Where the component is reusable IP, this value indicates if the customer has modified the behavior of the component. In most cases this field is `0x0`.

10.14.2.41 css600_cti Component Identification Register 0, CIDR0

The CIDR0 register is part of the set of component identification registers.

Attributes

Offset

`0x0FF0`

Type

Read-only

Reset

`0x0000000D`

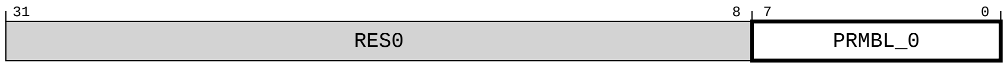
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-397: css600_cti_CIDR0



The following table shows the bit assignments.

Table 10-411: CIDR0 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0xD`	PRMBL_0	Read-only	Preamble. Returns `0x0D`.

10.14.2.42 css600_cti Component Identification Register 1, CIDR1

The CIDR1 register is part of the set of component identification registers.

Attributes

Offset

`0x0FF4`

Type

Read-only

Reset

`0x00000090`

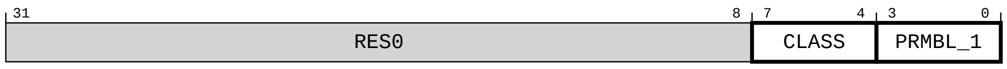
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-398: css600_cti_CIDR1



The following table shows the bit assignments.

Table 10-412: CIDR1 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:4]	`0b1001`	CLASS	Read-only	Component class. Returns `0x9`, indicating this is a CoreSight component.
[3:0]	`0b0000`	PRMBL_1	Read-only	Preamble. Returns `0x0`.

10.14.2.43 css600_cti Component Identification Register 2, CIDR2

The CIDR2 register is part of the set of component identification registers.

Attributes

Offset

`0x0FF8`

Type

Read-only

Reset

`0x00000005`

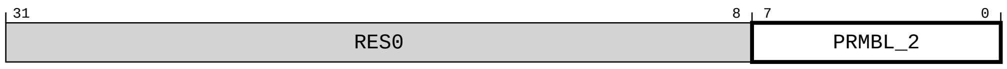
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-399: css600_cti_CIDR2



The following table shows the bit assignments.

Table 10-413: CIDR2 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0x5`	PRMBL_2	Read-only	Preamble. Returns `0x05`.

10.14.2.44 css600_cti Component Identification Register 3, CIDR3

The CIDR3 register is part of the set of component identification registers.

Attributes

Offset

`0x0FFC`

Type

Read-only

Reset

`0x000000B1`

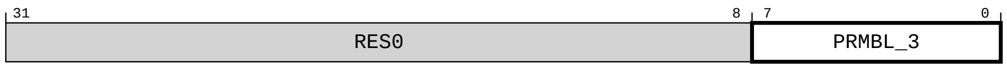
Width

32

Bit descriptions

The following image shows the bit assignments.

Figure 10-400: css600_cti_CIDR3



The following table shows the bit assignments.

Table 10-414: CIDR3 attributes

Bits	Reset value	Name	Type	Function
[31:8]	`0x0`	RES0	Read-only	Reserved bit or field with SBZP behavior
[7:0]	`0xB1`	PRMBL_3	Read-only	Preamble. Returns `0xB1`.

Appendix A Revisions

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

A.1 Revisions

Each table shows the technical differences between successive issues of the document.

Table A-1: Issue 0000-00

Change	Location
First release	-

Table A-2: Differences between issue 0000-00 and issue 0000-01

Change	Location
Corrected product name to SoC-600M	4.5 APB PADDRDBG31 adapter on page 36, 5.9.3.1 Signals of the trace out port on page 58

Table A-3: Differences between issue 0000-01 and issue 0100-00

Change	Location
Updated AHB-AP component	Including 3.2 Memory Access Ports on page 20
Updated Trace Memory Controller information	5.8 Trace Memory Controller on page 42
Updated Cortex-M3 and Cortex-M4 PIL PIDs	9.2.1 Cortex-M3 PIL CoreSight component identification on page 83, 9.3.1 Cortex-M4 PIL CoreSight component identification on page 86
Updated Programmers model information for `css600_dp`, `css600_apbap`, `css600_ahbap`, `css600_apv1adapter`, `css600_jtagap`, `css600_apbrom`, `css600_apbrom_gpr`, `css600_atbfunnel_prog`, `css600_atbreplicator_prog`, `css600_tmc_etb`, `css600_tpiu`, `css600_tsgen`, `css600_cti`	10. Programmers model on page 88

Table A-4: Differences between issue 0100-00 and issue 0101-00

Change	Location
Editorial updates	Throughout
Corrected and updated component versions and revisions	2.6 Component list on page 17
Updated TMC mode information	• 5.8 Trace Memory Controller on page 42 • 5.8.4.1 Architectural state machine on page 49 • 5.8.5.1 CB mode on page 55 • 5.8.5.2 SWF1 mode on page 56 • 5.8.5 Standard usage models on page 55
Included missing figures	5.9.9 Example configuration scenarios on page 61

Change	Location
• Added field type to assignments tables • Updated some reset values • Clarified Prot function • Corrected some RAZ/WI fields to RES0	10.1 Components programmers model on page 88