其他参考

**ARM出版物：**

• Arm® Architecture Reference Manual Armv7-A and Armv7-R edition (DDI 0406).

• Arm® Architecture Reference Manual Armv8, for Armv8-A architecture profile (DDI 0487).

• Arm® Cortex®-A53 MPCore Processor Advanced SIMD and Floating-point Extension Technical Reference Manual (DDI 0502).

• Arm® Cortex®-A Series Programmer’s Guide (DEN 0013).

• Arm® AMBA® AXI and ACE Protocol Specification AXI3, AXI4, and AXI4-Lite, ACE and ACE-Lite (IHI 0022).

• Arm® AMBA® APB Protocol Specification (IHI 0024).

• Arm® CoreSight™ Architecture Specification (IHI 0029).

• Arm® Debug Interface Architecture Specification, ADIv5.0 to ADIv5.2 (IHI 0031).

• Arm® AMBA® 4 ATB Protocol Specification (IHI 0032).

• Arm® Generic Interrupt Controller Architecture Specification (IHI 0048).

• Arm® ETM Architecture Specification, ETMv4 (IHI 0064).

• Low Power Interface Specification: Arm® Q-Channel and P-Channel Interfaces (IHI 0068).

需要licensees的文档：

• Arm® Cortex®-A53 MPCore Processor Cryptography Extension Technical Reference Manual (DDI 0501).

• Arm® Cortex®-A53 MPCore Processor Configuration and Sign-off Guide (DII 0281).

• Arm® Cortex®-A53 MPCore Processor Integration Manual (DIT 0036).

• Arm® AMBA® 5 CHI Protocol Specification (IHI 0050).

• Armv8 AArch32 UNPREDICTABLE behaviors.

其他出版物，第三方：

• ANSI/IEEE Std 754-2008, IEEE Standard for Binary Floating-Point Arithmetic.

注：

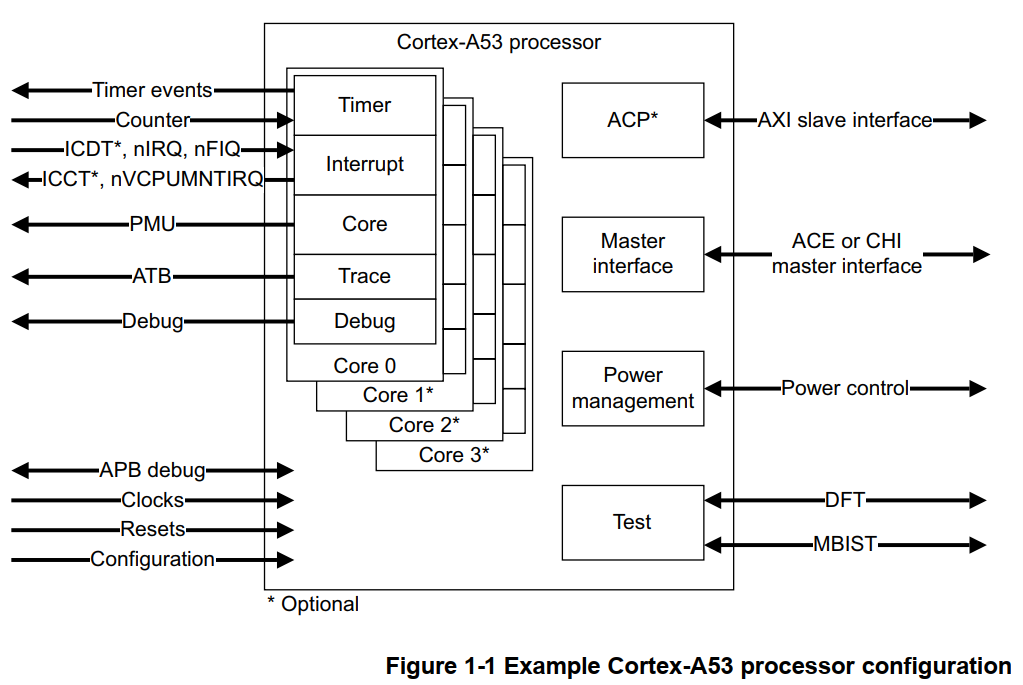
Arm floating-point terminology大部分基于早期的ANSI/IEEE Std 754-1985标准，详见Arm® Architecture Reference Manual Armv8, for Armv8-A architecture profile。

# 简介

## 关于Cortex-A53处理器

Cortex-A53 处理器是一款实现 Armv8-A 架构的中档低功耗处理器。 Cortex-A53 处理器具有一到四个core（每个core都有一个 L1 存储系统），和一个共享的 L2 cache。

图 1-1 显示了一个 Cortex-A53 MPCore 配置示例，该配置具有四个core和一个 ACE 或一个 CHI 接口。



## 规范

Cortex-A53遵从实现的规范有：

• Arm architecture.

• Interconnect architecture.

• Generic Interrupt Controller architecture on page 1-4.

• Generic Timer architecture on page 1-4.

• Debug architecture on page 1-4.

• Embedded Trace Macrocell architecture on page 1-4.

This TRM complements architecture reference manuals, architecture specifications, protocol

specifications, and relevant external standards. It does not duplicate information from these

sources.

### ARM架构

Cortex-A53 处理器实现了 Armv8-A 架构。这包括：

• 支持 AArch32 和 AArch64 执行状态（Execution state）。

• 在每个执行状态下支持所有异常级别EL0、EL1、EL2 和EL3。

• A32 指令集，以前称为 Arm 指令集。

• T32 指令集，以前称为Thumb 指令集。

• A64 指令集。

Cortex-A53 处理器支持以下架构扩展：

• 用于整数和浮点向量运算的可选高级SIMD浮点扩展。

注

— 高级 SIMD 架构、其相关实现和支持软件通常称为 NEON 技术。

— 要执行浮点运算，您必须实现高级 SIMD浮点扩展。 Armv8-A 架构中没有用于浮点的软件 API 库。

— 如果没有高级 SIMD，您将无法实现浮点。

• 可选的 Armv8 加密扩展。注意您不能在没有高级SIMD浮点的情况下实现加密扩展。

有关更多信息，请参阅 Arm® 架构参考手册 Armv8，了解 Armv8-A 架构配置文件。

### 互连架构

The Cortex-A53 bus interface natively supports one of:

• AMBA 4 ACE bus architecture. See the Arm® AMBA® AXI and ACE Protocol Specification AXI3, AXI4, and AXI4-Lite, ACE and ACE-Lite.

• AMBA 5 CHI bus architecture. See the Arm® AMBA® 5 CHI Protocol Specification.

### 通用中断控制器架构

Cortex-A53实现了通用中断控制器（Generic Interrupt Controller (GIC)） v4 架构。

Cortex-A53仅包含GIC CPU接口. 详见Arm® Generic Interrupt Controller Architecture Specification.

### 通用定时器架构

Cortex-A53实现了arm通用定时器（Generic Timer）架构。详见Arm® Architecture Reference Manual Armv8, for Armv8-A architecture profile.

### 调试架构

Cortex-A53实现了Armv8调试架构（Debug architecture）。

CoreSight Cross Trigger Interface (CTI)使能调试逻辑；

Embedded Trace Macrocell (ETM)和Performance Monitor Unit (PMU)互相连接并连接到其他CoreSight组件，详见：

• Arm® CoreSight™ Architecture Specification.

• Arm® Architecture Reference Manual Armv8, for Armv8-A architecture profile.

### 嵌入跟踪宏单元架构

ETM(Embedded Trace Macrocell)，Cortex-A53实现了ETMv4架构，详见Arm® ETM Architecture Specification, ETMv4.

## 特性

Cortex-A53 处理器包括以下特性：

• Armv8-A 架构指令集的完整实施，以及1.2.1的 Arm 架构中列出的架构选项。

• 具有大多数指令的对称双发射的顺序流水线。

• 带有内存管理单元 (MMU) 的哈佛1级(L1)存储系统。

• 提供集群（cluster）存储一致性的2级(L2)存储系统，可选地包括L2 cache。

## 接口

Cortex-A53 处理器具有以下外部接口：

• 实现ACE 或CHI 接口的存储器接口。

• 实现AXI slave接口的可选加速器一致性端口(ACP, Accelerator Coherency Port)。

• 实现APB slave接口的调试接口。

• 实现ATB 接口的跟踪接口。

• CTI。

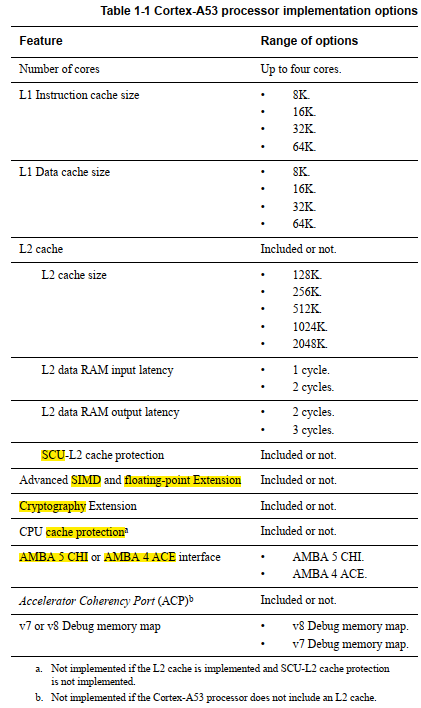
• 测试设计（DFT, Design for Test）。

• 内存内置自检 (MBIST, Memory Built-In Self-Test)。

• Q 通道，用于电源管理。

## 构建选项

Cortex-A53构建时可选项：



注意

• SCU 中的L1副本标签受CPU缓存保护。

• 没有高级SIMD就没有实现浮点的选项。

• 如果没有高级SIMD和浮点扩展，则无法实施加密扩展。

• 所有Core共享同一L2 Cache。

### 处理器配置

集群中的所有核心都具有相同的配置，这些配置是在构建配置期间确定的。这些配置不能由软件更改：

• 要么所有core都具有 L1 缓存保护，要么没有。

• 要么所有core都具有高级 SIMD 和浮点扩展，要么都没有。

• 要么所有core都有加密扩展，要么没有。

• 所有core必须具有相同大小的L1 高速缓存。

## 测试特征

Cortex-A53 处理器提供测试信号，支持使用 ATPG 和 MBIST 来测试处理器及其存储阵列。有关详细信息，请参阅附录 A 信号描述。

## 产品文档和设计流程

本节介绍 Cortex-A53 文档以及它们与以下设计流程的关系：

• 文档

• 设计流程

有关本节所述文档的更多信息，请参阅第 ix 页的补充阅读。有关相关架构标准和协议的信息，请参阅第 1-3 页的合规性。

### 文档

**Technical Reference Manual**

技术参考手册 (TRM) 描述了对 Cortex-A53 的功能和影响功能和行为的配置选项。在设计流程的所有阶段都需要它。在设计流程中做出的配置选择，意味着TRM中描述的某些某些功能已不相关。因此，如果您正在对 Cortex-A53 处理器进行编程，请联系：

• 实施者以确定：

— 实施的构建配置。

— 在实现 Cortex-A53 处理器之前执行了哪些集成（如果有）。

• 集成商以确定正在使用的设备的引脚配置。

有单独的 TRM 用于：

• 可选的高级 SIMD 和浮点扩展。

• 可选的加密扩展。

**Configuration and Sign-off Guide**

配置和签核指南 (CSG) 描述：

• 可用的构建配置选项和匹配的相关问题。

• 如何使用构建配置RTL (RTL, Register Transfer Level) 源文件。

• 如何集成 RAM 阵列。

• 如何运行测试向量。

* 对配置好的设计进行签署的流程。

Arm 产品可交付件包括参考脚本和有关使用它们来实现您的设计的信息。 Arm 提供的参考方法流程是示例参考实现。请联系您的 EDA 供应商以获得 EDA 工具支持。

CSG 是一本机密文档，仅供被许可人使用。

**Integration Manual**

集成手册 (IM) 描述了如何将 Cortex-A53 处理器集成到 SoC 中。

包括集成时必须tie off的引脚说明，Cortex-A53配置对集成的影响。

### 设计流程

Cortex-A53要达到可综合RTL交付，在它可应用于产品之前，必须经过以下过程：

实现： 实现者配置和综合 RTL 以产生硬宏单元。这包括将 RAM 集成到设计中。

集成： 集成器将宏单元连接到 SoC。这包括将其连接到存储系统和外围设备。

编程： 这是最后一个过程。系统程序员开发配置和初始化 Cortex-A53 处理器所需的软件，并测试所需的应用软件。

每个过程：

• 可以由不同的一方执行。

• 可以包括影响Cortex-A53 处理器的行为和特性的实施和集成选择。

最终设备的操作取决于：

构建配置： 实施者选择影响 RTL 源文件如何预处理的选项。这些选项通常包括或排除影响所产生宏小区的面积、最大频率和特征中的一项或多项的逻辑。

配置输入： 集成器通过将输入绑定到特定值来配置 Cortex-A53 处理器的某些功能。在进行任何软件配置之前，这些配置会影响启动行为。他们还可以限制软件可用的选项。

软件配置： 程序员通过将特定值编程到寄存器中来配置 Cortex-A53 处理器。这会影响处理器的行为。

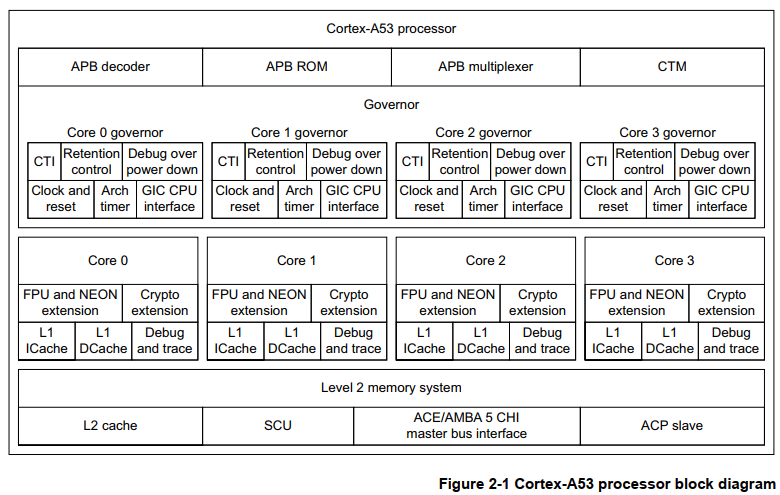
注意

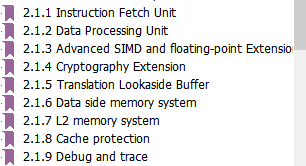
本手册涉及适用于构建配置选项的实现定义的功能。引用包含的功能意味着已选择适当的构建和引脚配置选项。引用启用的功能意味着该功能也已由软件配置。

# 功能描述

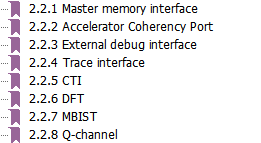
## Cortex-A53的功能

功能图概览





## 接口



## 时钟和复位

### 时钟

Cortex-A53 处理器具有单个时钟输入 CLKIN。 Cortex-A53 处理器和 SCU 中的所有core都使用CLKIN的分发信号做为时钟。

Cortex-A53 处理器具有以下时钟使能信号：

• PCLKENDBG。

• ACLKENM。

• ACLKENS。

• SCLKEN。

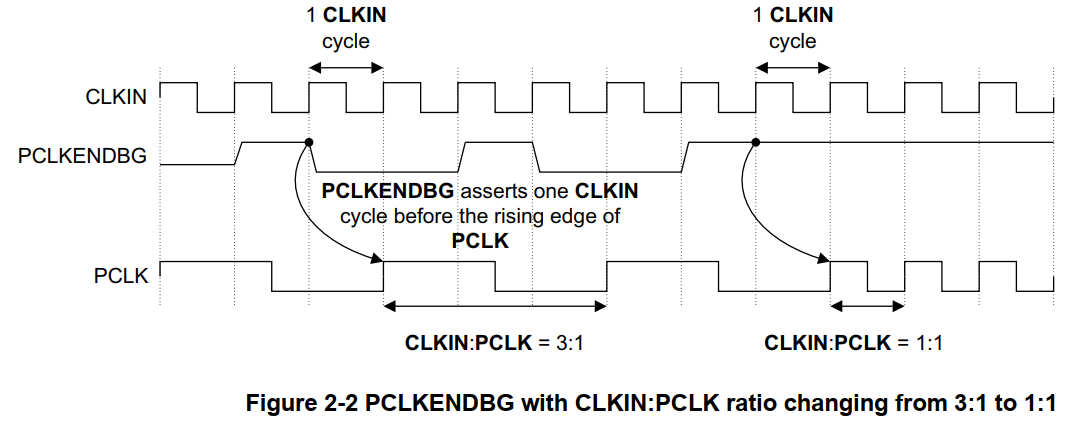
• ATCLKEN。

• CNTCLKEN。

#### PCLKENDBG

处理器包括一个 APB 接口，用于访问调试和性能监控寄存器。该接口在内部由CLKIN 驱动。提供单独的使能信号PCLKENDBG以使外部APB总线能够以较低频率驱动，该频率必须是 CLKIN 的整数比。如果系统中的调试基础设施需要与处理器时钟完全异步，您可以使用同步组件将外部 AMBA APB连接到处理器。

下图示例，用PCLKENDBG，将CLKIN和PCLK的频率比，由3:1调整为1:1。



注意

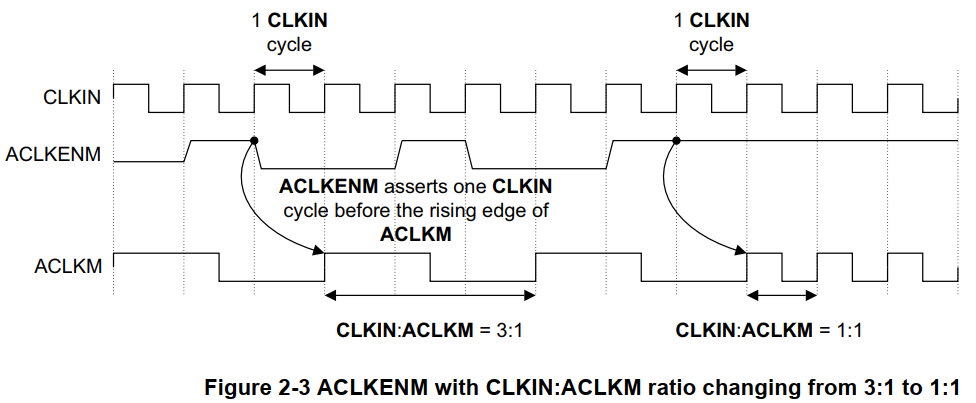
图 2-2 显示了调试APB 时钟PCLK 和 PCLKENDBG 之间的时序关系，其中 PCLKENDBG 在 PCLK 上升沿之前断言一个时钟周期。保持PCLK和PCLKENDBG之间的关系很重要。

#### ACLKENM

此信号仅在主接口配置为使用 ACE 协议时出现。主接口支持 CLKIN 频率的整数比，例如 1:1、2:1、3:1。这些比率通过外部时钟使能信号进行配置。在所有情况下，AXI 传输都保持同步。 ACE 主接口包括 ACLKENM 时钟使能信号。

ACLKENM 在外部 ACE 时钟信号 ACLKM 的上升沿之前断言一个 CLKIN 周期。如果更改 CLKIN 与 ACLKM 的频率比，则必须相应更改 ACLKENM。

图 2-3 显示了 ACLKENM 的时序示例，它将 CLKIN和ACLKM 的频率比从 3:1 更改为 1:1。



注意

• 图 2-3 显示了AXI主时钟ACLKM 和 ACLKENM 之间的时序关系，其中ACLKENM在 ACLKM 的上升沿之前断言一个时钟周期。保持ACLKM和ACLKENM之间的关系很重要。

• 如果在更改时钟频率时可能发生某些物理影响，ARM建议仅在处理器的 STANDBYWFIL2 输出置位时更改时钟比率。

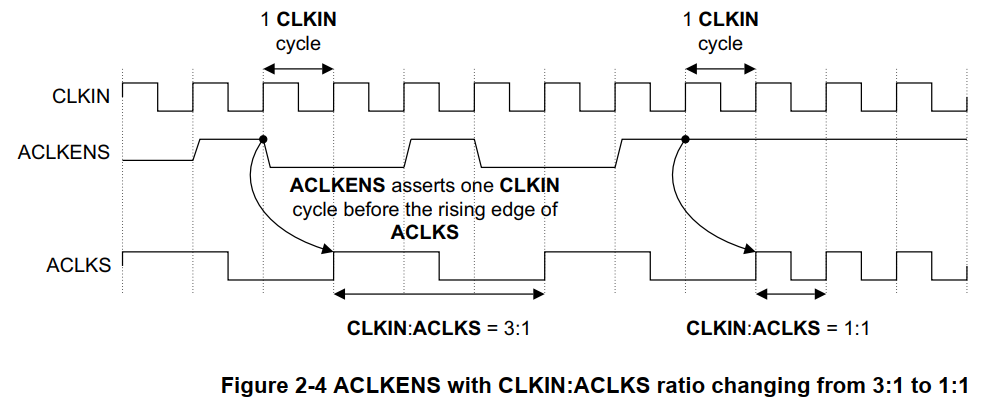
• 如果Cortex-A53 处理器配置为包含ACE 接口，则输入信号ACLKENM 存在于该处理器中。

#### ACLKENS

该信号仅在处理器配置有 ACP 从接口时出现。从接口支持 CLKIN 频率的整数比，例如 1:1、2:1、3:1。这些比率通过外部时钟使能信号进行配置。在所有情况下，AXI 传输都保持同步。 ACP 从接口包括 ACLKENS 时钟使能信号。

ACLKENS 在外部 ACP 时钟信号 ACLKS 的上升沿之前断言一个 CLKIN 周期。如果更改 CLKIN 与 ACLKS 的频率比，则必须相应更改 ACLKENS。

第 2-11 页的图 2-4 显示了 ACLKENS 的时序示例，它将 CLKIN和ACLKS 的频率比从 3:1 更改为 1:1。



注意

• 图 2-4 显示了 AXI 从时钟ACLKS 和 ACLKENS 之间的时序关系，其中 ACLKENS 在 ACLKS 的上升沿之前断言一个时钟周期。保持 ACLKS 和 ACLKENS 之间的关系很重要。

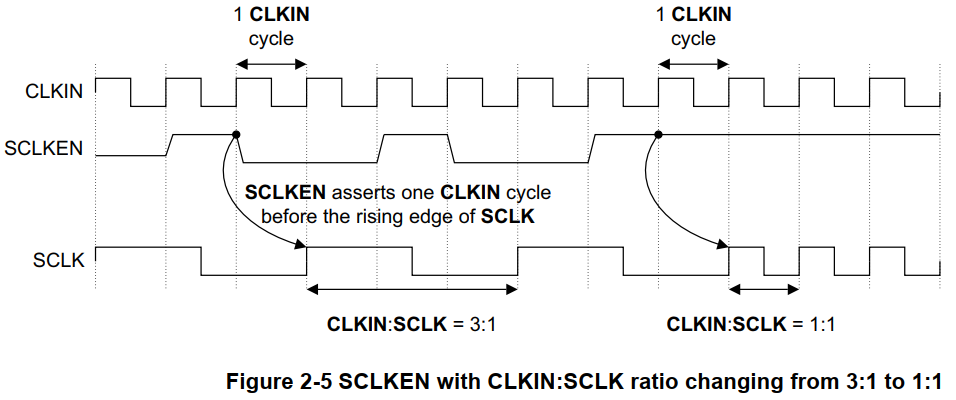
• 如果在更改时钟频率时可能发生某些物理影响，ARM 建议仅在处理器的 STANDBYWFIL2 输出置位时更改时钟比率。

• 如果Cortex-A53 处理器配置为包含ACP 接口，则输入信号ACLKENS 存在于该处理器中。

#### SCLKEN

此信号仅在主接口配置为使用 CHI 协议时出现。 SCU 接口支持 CLKIN 频率的整数比，例如 1:1、2:1、3:1。这些比率通过外部时钟使能信号进行配置。在所有情况下，CHI 传输都保持同步。 CHI 主接口包括 SCLKEN 时钟使能信号。

图 2-5 显示了将 CLKIN 与 SCLK 频率比从 3:1 更改为 1:1 的 SCLKEN 时序示例。



注意

• 图 2-5 显示了 CHI 时钟SCLK 和 SCLKEN 之间的时序关系，其中 SCLKEN 在 SCLK 上升沿之前断言一个 CLKIN 周期。保持 SCLK 和 SCLKEN 之间的关系很重要。

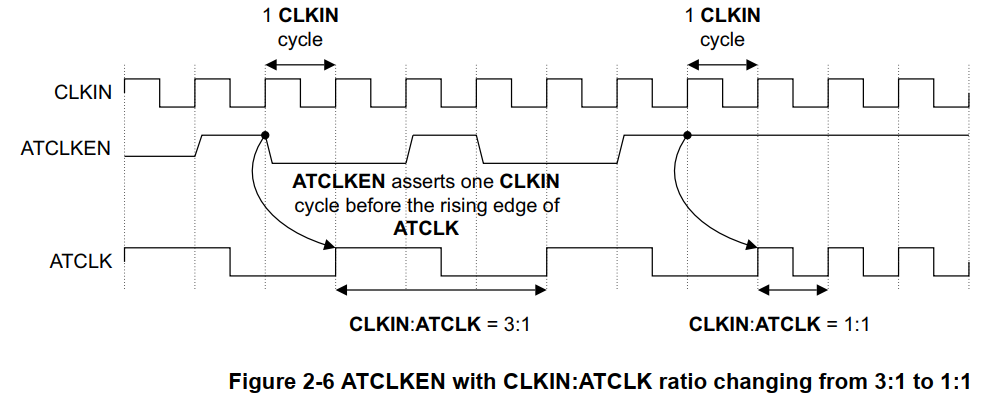
• 如果在更改时钟频率时可能发生某些物理影响，ARM 建议仅在处理器的 STANDBYWFIL2 输出置位时更改时钟比率。

• 如果Cortex-A53 处理器配置为包含CHI 接口，则输入信号SCLKEN 存在于该处理器中。

#### ATCLKEN

ATB 接口是一个同步接口，可以使用 ATCLKEN 信号以等于或低于主处理器时钟 CLKIN 的任何整数倍运行。例如，CLKIN 与 ATCLK 的频率比可以是 1:1、2:1 或 3:1，其中 ATCLK 是 ATB 总线时钟。 ATCLKEN 在 ATCLK 上升沿之前断言一个 CLKIN 周期。如果更改 CLKIN 与 ATCLK 的频率比，则必须相应更改 ATCLKEN。

图 2-6 显示了 ATCLKEN 的时序示例，它将 CLKIN 与 ATCLK 的频率比从 3:1 更改为 1:1。



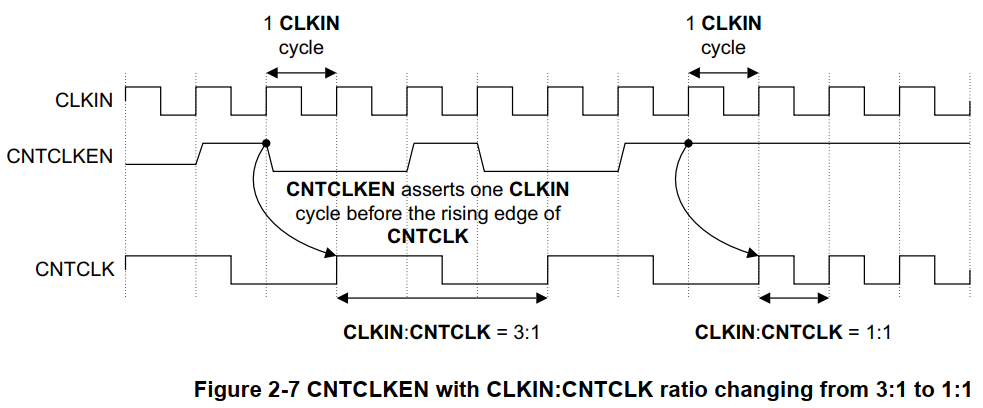
注意

图 2-6 显示了ATB时钟ATCLK 和 ATCLKENDBG 之间的时序关系，其中 ATCLKENDBG 在 ATCLK 上升沿之前断言一个时钟周期。保持 ATCLK 和 ATCLKENDBG 之间的关系很重要。

#### CNTCLKEN

CNTVALUEB 是一个同步的 64 位二进制编码计数器值，可以使用 CNTCLKEN 信号以等于或低于主处理器时钟 CLKIN 的任何整数倍运行。例如，您可以将 CLKIN 与 CNTCLK 频率比设置为 1:1、2:1 或 3:1，其中 CNTCLK 是系统计数器时钟。 CNTCLKEN 在 CNTCLK 上升沿之前断言一个 CLKIN 周期。

第 2-13 页的图 2-7 显示了 CNTCLKEN 的时序示例，它将 CLKIN 与 CNTCLK 的频率比从 3:1 更改为 1:1。



注意

图 2-7 显示了系统计数器时钟 CLKIN 和 CNTCLKEN 之间的时序关系，其中 CNTCLKEN 在 CNTCLK 上升沿之前断言一个时钟周期。保持 CNTCLK 和 CNTCLKEN 之间的关系很重要。

### 输入同步

Cortex-A53对下属输入信号做了同步:

• nCORERESET.

• nCPUPORESET.

• nFIQ.

• nIRQ.

• nL2RESET.

• nMBISTRESET.

• nPRESETDBG.

• nREI.

• nSEI.

• nVFIQ.

• nVIRQ.

• nVSEI.

• CLREXMONREQ.

• CPUQREQn.

• CTICHIN.

• CTICHOUTACK.

• CTIIRQACK.

• DBGEN.

• EDBGRQ.

• EVENTI.

• L2FLUSHREQ.

• L2QREQn.

• NEONQREQn.

• NIDEN.

• SPIDEN.

• SPNIDEN.

SoC 可以将这些信号作为异步输入。所有其他外部信号必须与 CLKIN 同步。

注意

同步CTICHIN输入信号仅在CISBYPASS输入信号置低时使用。如果CISBYPASS信号被置为高电平，则不使用CTICHIN同步器，并且SoC必须将CTICHIN同步到CLKIN。

### 复位

Cortex-A53 处理器具有以下低电平有效复位输入信号：

|  |  |  |
| --- | --- | --- |
| nCPUPORESET[CN:0] | 其中CN是core数减一。这些主要的冷复位信号初始化相应core中的所有可复位寄存器，包括调试寄存器和ETM寄存器。 | 接reset control |
| nCORERESET[CN:0] | 这些复位信号初始化相应core中的所有可复位寄存器，不包括调试寄存器和ETM寄存器。 |
| nPRESETDBG | 这个单一的、集群范围的信号复位连接到外部PCLK域的集成 CoreSight 组件，例如调试逻辑。 |
| nL2RESET | 这个单一的、集群范围的信号复位L2存储系统中的所有可复位寄存器和SCU中的逻辑。 |
| nMBISTRESET | 外部MBIST控制器可以使用此信号来复位整个SoC。nMBISTRESET信号复位集群中的所有复位寄存器，以便进入和退出MBIST模式。 |

所有这些复位都可以是异步：

• asserted，从高到低。

• deasserted，从低到高。

Cortex-A53 处理器内的复位同步撤离逻辑确保复位解除对于所有可复位寄存器是同步的。复位assertion不需要处理器时钟，但复位deassertion必须同步到处理器时钟，以确保复位同步。

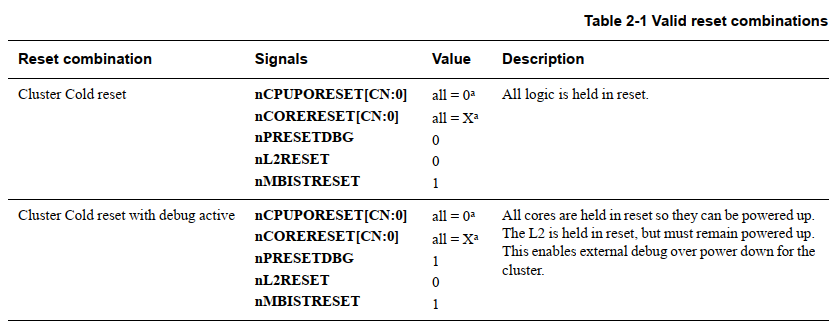
通常，您只需将复位信号保持三个处理器时钟周期有效，复位即可生效。但是，在以下情况下，必须将复位信号保持在低电平，直到电源恢复并且单元或处理器准备好使复位生效：

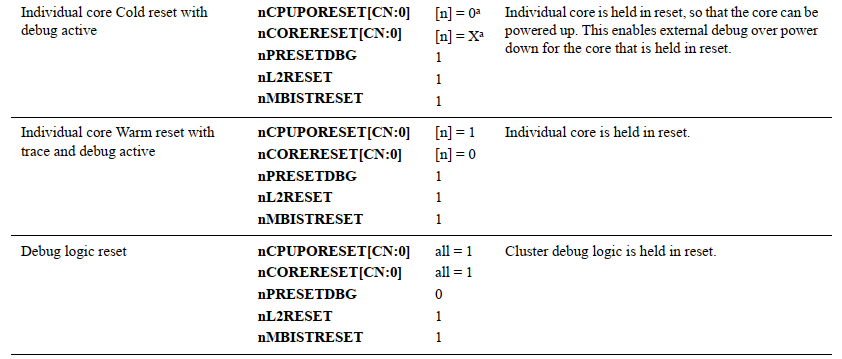
• 正在进行复位的Core的高级SIMD和FP处于retention状态。

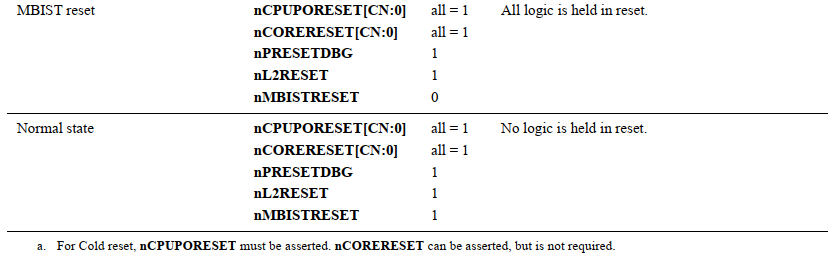
• 正在复位的core处于retention状态。

这是系统构建者决定的，因为retetion退出所用的时间和电源控制器的行为因合作伙伴和实施而异。

表 2-1 描述了有效的复位信号组合。所有其他的复位信号组合都是非法的。在表中，n 表示被复位的core。



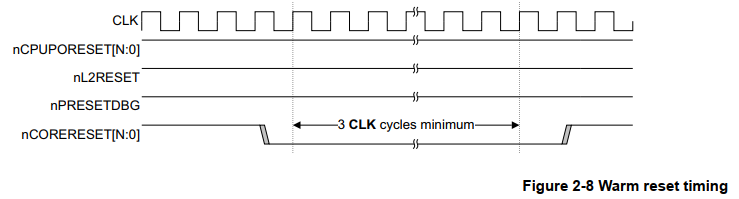




**热复位 Warm reset**

热复位初始化单个core中的所有逻辑，除了CLK域中的调试和ETM逻辑。在热复位序列期间保留所有断点和观察点。

下图显示了 Cortex-A53 处理器的热复位序列。



单个core热复位初始化单个core中的所有逻辑，除了其调试、ETM、断点和观察点逻辑。保留该核心的断点和观察点。您必须先应用正确的顺序，然后再将热复位应用到该core。

对于单个core热复位：

• 您必须在core断电序列中应用步骤 1 到 6，请参阅第 2-22 页上的单个core关闭模式，并等到STANDBYWFI asserted，表明core处于空闲状态，然后再为此core assert nCORERESET。

• 该core的nCORERESET 必须置位至少3 个CLK 周期。

• 当任何单个core处于活动状态时，nL2RESET 不可assert。

• 当任何单个core在正常操作模式下主动调试时，nPRESETDBG 不可assert。

注意

如果CPU Q-channel接口的core正在使用dynamic retention，在应用 nCORERESET 之前，core必须处于quiescent状态，且STANDBYWFI asserted，并且CPUQREQn、CPUQACCEPTn 和 CPUQACCEPT 必须为低。

**WARMRSTREQ和DBGRSTREQ**

ARMv8-A 架构提供了一种机制来配置处理器在热复位后在EL3上使用 AArch32 还是 AArch64。当RMR或RMR\_EL3寄存器中的复位请求位设置1时，处理器会置位 WARMRSTREQ 信号，并且SoC复位控制器可以使用该信号来触发core的热复位并更改AArch64/AArch32状态。 RMR 或 RMR\_EL3 寄存器中的 AA64 位选择下一次热复位时EL3的执行架构。

请参阅 ARM® 架构参考手册 ARMv8，了解 ARMv8-A 架构配置文件，了解有关建议使用的代码序列的信息，以请求热复位。

必须在core断电步骤中应用步骤1到6（单core关闭），并等到STANDBYWFI置位表明处理器处于空闲状态，然后再为该core置位nCORERESET。 nCORERESET必须满足热复位部分中描述的时序要求。

## 电源管理

Power management

Cortex-A53 处理器提供控制动态和静态功耗的机制和支持。 Cortex-A53 处理器中的各个core支持四个主要级别的电源管理。本节介绍：

• 电源域(Power domains)。

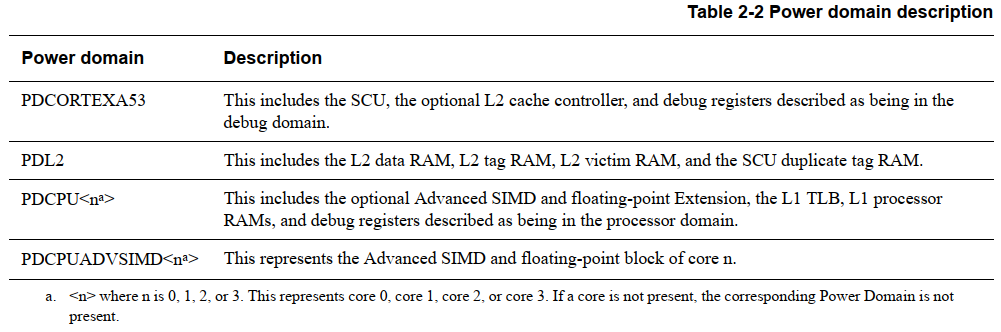
• 电源模式(Power modes)。

• 使用WFE或SEV进行事件通信。

• 与电源管理控制器的通信。

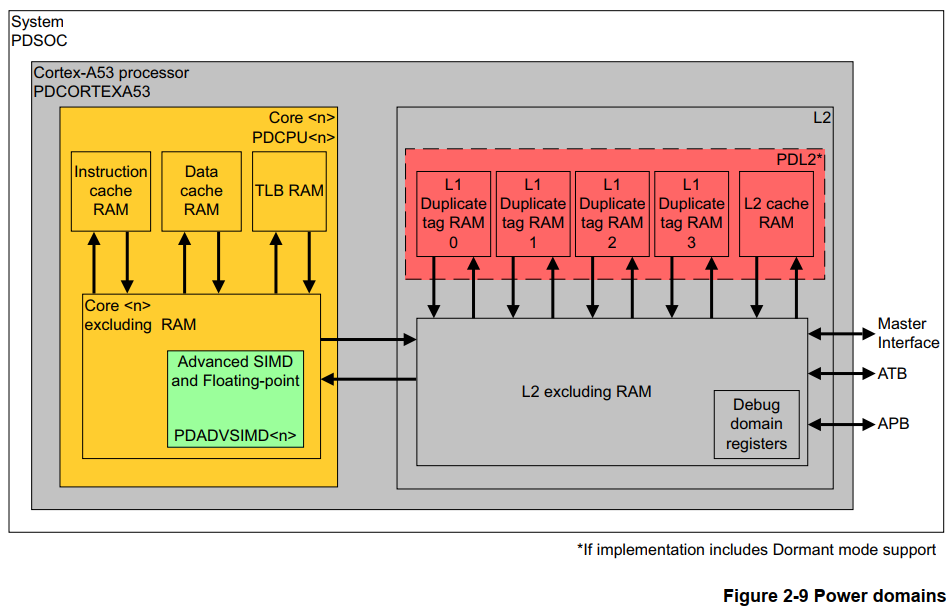
### 电源域

Cortex-A53支持的电源域：



即使所有core都断电，单独的PDCORTEXA53和PDL2电源域也可以保持活动状态。这意味着 Cortex-A53 处理器可以继续接受来自外部设备的Snoops访问 L2 cache。

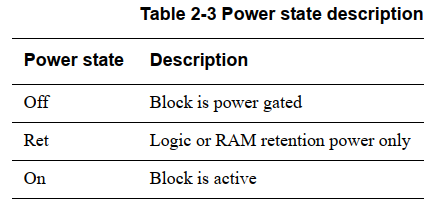
第 2-18 页上的图 2-9 显示了嵌入在片上系统 (SoC) 电源域中的域示例。



### 电源模式

电源域可以独立控制，以提供上电(powered-up)和断电(powered-down)域的不同组合。但是，只有一些上电和断电域组合有效并受支持。

第 2-19 页的表 2-4 和第 2-19 页的表 2-5 显示了 Cortex-A53 处理器支持的电源域状态。使用的术语在表 2-3 中定义。



注意

未在表2-4和表2-5中显示的状态不受支持且不得出现。

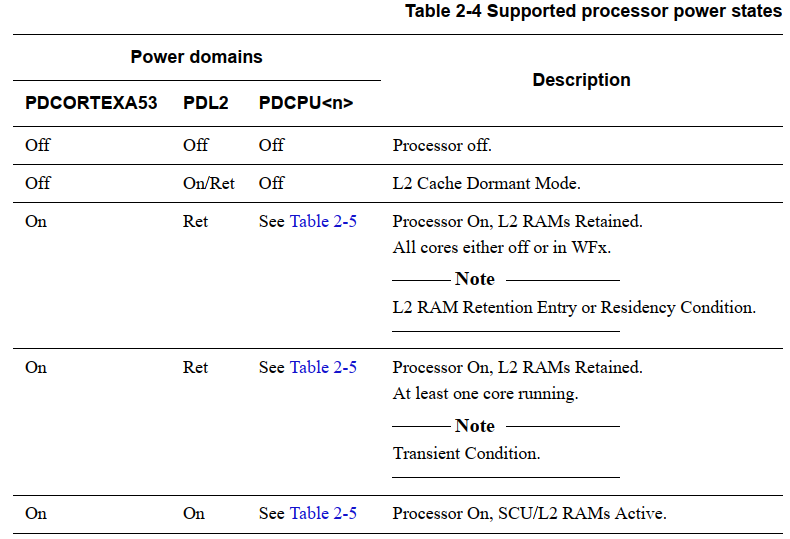
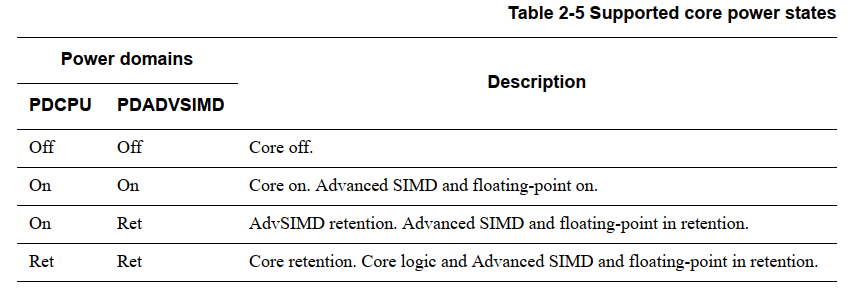


表2-5描述了各个core支持的电源域状态。每个core中的电源域状态独立于所有其他core。



您必须遵循以下部分中描述的动态电源管理以及上电和断电顺序。与这些序列的任何偏差都可能导致不可预测的结果。

支持如下电源模式：

• Normal state

• Standby state

• Individual core shutdown mode

• Cluster shutdown mode without system driven L2 flush

• Cluster shutdown mode with system driven L2 flush

• Dormant mode

• Retention state

#### Normal state

这是所有处理器功能都可用的正常操作模式。 Cortex-A53 处理器使用门控时钟和门来禁用未使用的功能块的输入。只有用于执行操作的逻辑会消耗动态功耗。

#### Standby state

以下部分描述了进入待机状态的方法：

• Core Wait for Interrupt，core等待中断。

• Core Wait for Event，core等待事件。

• L2 Wait for Interrupt，L2等待中断。

**Core WFI**

等待中断是ARMv8-A架构的一项功能，将core置于低功耗状态，通过禁用Core中的大部分时钟，但保持core通电。除了使core能够从WFI低功耗状态唤醒的逻辑上的小动态功耗开销外，这仅减少了从静态漏电流到电源迁移的功耗。

软件指示core可以通过执行WFI指令进入WFI低功耗状态。

当core在执行**WFI指令**时，core会等待core中的所有指令退出，然后再进入空闲或低功耗状态。WFI指令确保按程序顺序在WFI指令之前发生的所有显式存储访问都已退出。例如，WFI 指令确保以下指令从L2存储系统接收到所需的数据或响应：

• Load指令。

• Cache和TLB 维护操作。

• Store exclusive指令。

此外，WFI指令确保存储指令已更新缓存或已发到SCU。

当core处于WFI低功耗状态时，当检测到以下任何事件时，core中的时钟会暂时启用，而不会导致core退出WFI低功耗状态：

• snoop请求必须由core L1 Data cache处理。

• Cache或TLB维护操作必须由core L1 Instruction cache、data cache或 TLB 处理。

• APB访问core电源域中的调试或跟踪寄存器。

当core检测到复位或WFI唤醒事件之一时，会退出WFI低功耗状态，如ARM® 架构参考手册 ARMv8 中所述，适用于 ARMv8-A 架构配置文件。

在进入WFI低功耗状态时，该core的STANDBYWFI 被置位。STANDBYWFI的置位保证core处于空闲和低功耗状态。即使core中的时钟因L2 snoop请求、cache或 TLB 维护操作或 APB 访问而临时启用，STANDBYWFI 也会继续置位。

注意

STANDBYWFI 并不表示处理器启动的 L2 存储系统事务已完成。所有 Cortex-A53 处理器实现都包含 L2 存储系统。这包括没有 L2 缓存的实现。

**Core WFE**

等待事件 (WFE) 是 ARMv8-A 架构的一项功能，可用于基于事件的锁定机制，通过禁用core中的大部分时钟同时保持core供电，从而将core置于低功耗状态。除了使core能够从WFE低功耗状态唤醒的逻辑上的小动态功耗开销外，这仅减少了从静态漏电流到电源迁移的功耗。

core通过执行**WFE指令**进入WFE低功耗状态。在执行WFE指令时，core会等待core中的所有指令完成，然后再进入空闲或低功耗状态。

如果设置了事件寄存器，则执行WFE不会导致进入待机状态，但会清除事件寄存器。

当core处于WFE低功耗状态时，当检测到以下任何事件时，core中的时钟会临时启用，而不会导致core退出 WFE 低功耗状态：

• snoop请求必须由core L1 Data cache处理。

• Cache或TLB维护操作必须由core L1 Instruction cache、data cache或 TLB 处理。

• APB访问驻留在core电源域中的调试或跟踪寄存器。

当core检测到复位、EVENTI 输入信号的置位或WFE唤醒事件之一时，将退出WFE低功耗状态，如 ARM® 架构参考手册 ARMv8 中所述，用于 ARMv8-A 架构配置文件。

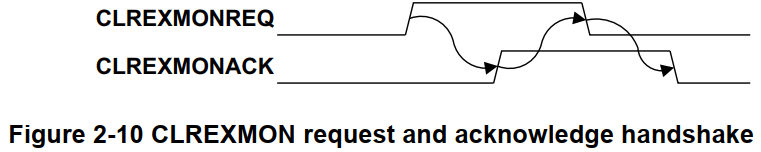
在进入WFE低功耗状态时，该core的STANDBYWFE被置位。STANDBYWFE的置位保证core处于空闲和低功耗状态。即使core中的时钟由于L2 snoop请求、cache或 TLB 维护操作或 APB 访问而临时启用，STANDBYWFE也会继续置位。

**CLREXMON 请求和确认信号**

当CLREXMONREQ输入被置位时，它发出清除外部全局排他监视器的信号，并充当集群中所有core的WFE唤醒事件。

CLREXMONREQ 信号具有相应的CLREXMONACK响应信号。这形成了一个标准的 2 线 4 阶段握手，可用于跨core和system之间的电压和频率边界发出信号。

图 2-10 显示了 CLREXMON 请求和确认握手。当请求信号被置位时，它会继续保持直到收到确认。当请求被置低时，确认可以置低。



**L2 WFI**

当所有core都处于WFI低功耗状态时，所有core共有的共享L2存储系统逻辑也可以进入WFI低功耗状态。

仅当满足特定要求并应用以下序列时，才能进入L2 WFI低功耗状态：

• 所有core都处于 WFI 低功耗状态，因此，每个core的STANDBYWFI输出都被置位。判断所有core STANDBYWFI 输出保证所有core都处于空闲和低功耗状态。除少量时钟唤醒逻辑外，core中的所有时钟都被禁用。

• 如果配置了ACE，则SoC 将输入引脚ACINACTM置位来使AXI master接口空闲。这表明不会从外部存储系统发来snoop请求。

• 如果配置了CHI 接口，则 SoC 将输入引脚SINACT置位来使CHI master接口空闲。这表明不会从外部存储系统发来snoop请求。

• 如果配置了ACP接口，则 SoC将输入引脚AINACTS置位来使ACP 接口空闲。这表明 SoC 在 ACP 接口上不再发送事务。

当 L2 存储系统完成AXI或CHI接口的未完成的事务后，它可以进入L2 WFI 低功耗状态。在进入 L2 WFI 低功耗状态时，STANDBYWFIL2 被置位。 STANDBYWFIL2 的置位保证了 L2 存储系统是空闲的并且不接受新的事务。

发生以下事件之一时退出 L2 WFI 低功耗状态：

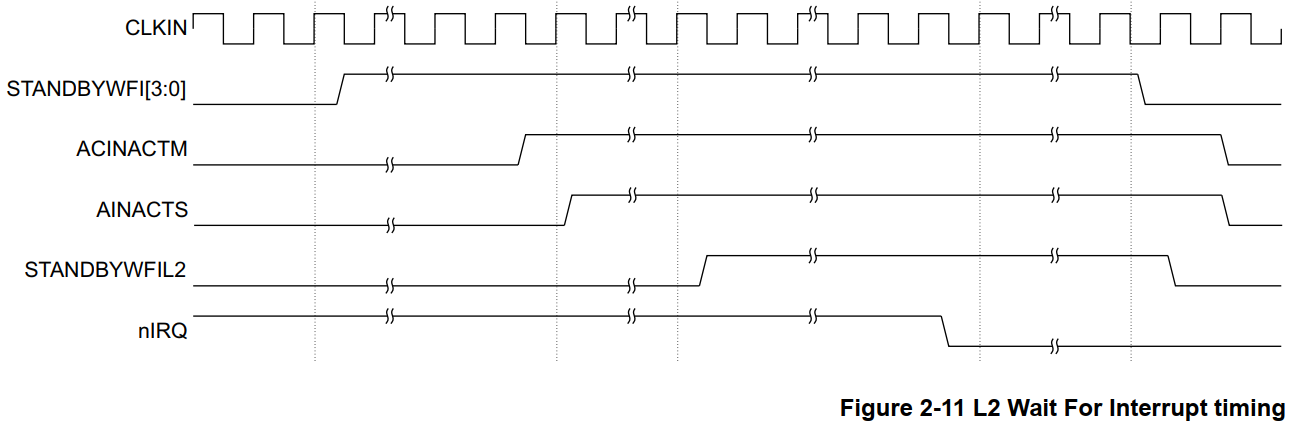
• 物理 IRQ 或 FIQ 中断。

• 调试事件。

• 上电或热复位。

当一个core退出 WFI 低功耗状态时，该core的 STANDBYWFI 被置低。当 L2 存储器系统逻辑退出 WFI 低功耗状态时，STANDBYWFIL2 被置低。 SoC 必须继续保持ACINACTM 或 SINACT置位，直到STANDBYWFIL2 取消置位。

图 2-11 显示了 4 核配置的 L2 WFI 时序。



#### Individual core shutdown mode

在此模式下，单个core的PDCPU电源域将关闭，所有状态都将丢失。

有关 Cortex-A53 处理器的完全关闭，包括使用单核的实现，请参阅第 2-24 页的无系统驱动 L2 刷新的集群关闭模式和第 2-24 页的使用系统驱动的 L2 刷新的集群关闭模式。

要使一个core断电，实现时必须将core放在单独控制的电源上。此外，必须在整个集群断电时将core的输出钳制为良性值，以指示core处于空闲状态。

要关闭core，请应用以下序列：

1. 通过清除 SCTLR.C 位或 HSCTLR.C 位（如果处于Hyp模式）来禁用data cache。这可以防止更多的data cache分配，并将cacheable属性更改为Normal Non-cacheable属性。随后的Load和store不会访问 L1 或 L2 cache。

2. Clean L1 Data cache的所有数据并使其无效。此core的L2 副本snoop tag RAM现为空。这可以防止来自集群中其他核心的任何新的数据缓存侦听或数据缓存维护操作被发布到该核心。

3. 通过清除CPUECTLR.SMPEN位来禁用与集群中其他core的数据一致性。清除SMPEN 位可以通过阻止core接收集群中其他core广播的cache或TLB 维护操作来使core脱离一致性。

4. 执行ISB指令以确保之前步骤中的所有寄存器更改都已提交。

5. 执行一条DSB SY指令，确保在SMPEN位被清除之前,集群设备中的任何core发出的所有cache、TLB和分支预测器维护操作都已完成。

6. 执行WFI 指令并等待直到 STANDBYWFI输出置位，以指示core处于空闲和低功耗状态。

7. 置低 DBGPWRDUP 为低电平。这可以防止对core的任何外部调试访问。

8. 激活核心输出钳位。

9. 置低nCPUPORESET LOW

10. 从PDCPU电源域中移除电源。

要为core加电，请应用以下顺序：

1. 置位 nCPUPORESET LOW。确保 DBGPWRDUP 保持低电平以防止对core进行任何外部调试访问。

2. 给PDCPU电源域上电。保持信号nCPUPORESET和DBGPWRDUP的状态为低。

3. 释放输出钳位

4. 复位撤销。

5. 将SMPEN位设置为1以启用对core的snoop。

6. 置位DBGPWRDUP HIGH以允许外部调试访问core。

7. 如果需要，使用软件将core的状态恢复到掉电前的状态。

#### Cluster shutdown mode without system driven L2 flush

这是 PDCORTEXA53、PDL2 和 PDCPU 电源域关闭且所有状态都丢失的模式。在本节中，lead core定义为最后一个关闭的core，或第一个打开的core。

要关闭集群，请按以下流程执行：

1. 确保所有non-lead cores都处于关闭模式，请参阅第2-22 页的单个core关闭模式。

2. 按照第2-22 页的独立core关闭模式中的步骤 1 至 2。

3. 如果配置了ACP接口，请确保连接到该接口的任何master设备不发送新事务，然后将AINACTS置位。

4. clean&invalid所有L2 Data cache。

5. 按照第 2-22 页的独立core关闭模式中的步骤 3 到 10。

6. 在ACE配置中，将ACIACTM置位，或者在CHI配置中，将SINACT置位。然后等到 STANDBYWFIL2 输出置位以指示 L2 存储系统处于空闲状态。所有 Cortex-A53 处理器实现都包含 L2存储系统，包括没有L2 cache的实现。

7. 激活集群输出钳位。

8. 断开 PDCORTEXA53 和 PDL2 电源域的电源。

注意

对于设备断电，在所有non-lead-core上的相同步骤之后，必须在lead-core上进行所有操作。

要启动集群，请应用以下顺序：

1. 对于集群中的每个core，置位 nCPUPORESET LOW。

2. nL2RESET置低并保持L2RSTDISABLE为低电平。

3. 为PDCORTEXA53和PDL2域供电，同时保持步骤 1. 和 2. 中描述的信号为低电平。

4. 释放集群输出钳位。

5. 继续正常的冷复位流程。

#### Cluster shutdown mode with system driven L2 flush

这是 PDCORTEXA53、PDL2 和 PDCPU 电源域关闭且所有状态都丢失的模式。

要关闭集群，请应用以下顺序：

1. 确保所有core都处于关闭模式，请参阅第 2-22 页的单个core关闭模式。

2. SoC 发出 AINACTS 信号以使 ACP 空闲。防止 ACP 事务在发生硬件缓存刷新时在 L2 缓存中分配新条目。

3. 将L2FLUSHREQ置为高。

4. 保持L2FLUSHREQ 为高电平，直到L2FLUSHDONE 被置位。

5. 置低L2FLUSHREQ。

6. 在ACE配置中，将ACIACTM置位，或者在CHI配置中，将SINACT置位。然后等到 STANDBYWFIL2 输出置位以指示 L2 存储系统处于空闲状态。所有 Cortex-A53 处理器实现都包含 L2存储系统，包括没有L2 cache的实现。

7. 激活集群输出钳位。

8. 断开 PDCORTEXA53 和 PDL2 电源域的电源。

注意

对于设备断电，在所有non-lead-core上的相同步骤之后，必须在lead-core上进行所有操作。

要启动集群，请应用以下顺序：

1. 对于集群中的每个core，置位 nCPUPORESET LOW。

2. nL2RESET置低并保持L2RSTDISABLE为低电平。

3. 为PDCORTEXA53和PDL2域供电，同时保持步骤1和2中描述的信号为低电平。

4. 释放集群输出钳位。

5. 继续正常的冷复位流程。

#### Dormant mode

可选地，集群中支持休眠模式。在此模式下，所有core和 L2 控制逻辑都断电，而 L2 cache RAM 通电并保持状态。在休眠模式下保持通电的RAM块是：

• L2 tag RAMs。

• L2 data RAMs。

• L2 victim RAM。

要支持休眠模式，必须确保：

• L2 cache RAM 位于单独的电源域中。

• 将L2 cache RAM 的所有输入钳制为良性值。这可以避免在core和L2控制电源域进入和退出断电状态时损坏数据。

在进入休眠模式之前，必须将集群的架构状态（不包括保持通电的 L2 cache RAM 的内容）保存到外部存储器。

作为从休眠模式退出到正常状态的一部分，SoC必须执行冷复位流程。 SoC 必须置位复位信号，直到电源恢复。电源恢复后，集群退出冷复位流程，必须恢复架构状态。

要进入休眠模式，应用以下序列：

1. 通过清除 SCTLR.C 位或 HSCTLR.C 位（如果处于Hyp模式）来禁用data cache。这可以防止更多的data cache分配，并将cacheable属性更改为Normal Non-cacheable属性。随后的Load和store不会访问 L1 或 L2 cache。

2. Clean L1 Data cache的所有数据并使其无效。此core的L2 副本snoop tag RAM现为空。这可以防止来自集群中其他核心的任何新的数据缓存侦听或数据缓存维护操作被发布到该核心。

3. 通过清除CPUECTLR.SMPEN位来禁用与集群中其他core的数据一致性。清除SMPEN 位可以通过阻止core接收集群中其他core广播的cache或TLB 维护操作来使core脱离一致性。

4. 如果需要，保存架构状态。这些状态保存操作必须确保发生以下情况：

• 保存所有ARM寄存器，包括CPSR和SPSR。

• 保存所有系统寄存器。

• 保存所有调试相关状态。

5. 执行 ISB 指令以确保之前步骤中的所有寄存器更改都已提交。

6. 执行一条DSB指令，确保在SMPEN位被清除之前集群中任何core发出的所有cache、TLB和分支预测器维护操作都已完成。此外，这可确保所有状态保存都已完成。

7. 执行WFI指令并等到STANDBYWFI输出置位，以指示core处于空闲和低功耗状态。

8. 对所有core重复上述步骤，并等待所有STANDBYWFI 输出被置位。

9. 如果配置了ACP接口，请确保连接到该接口的任何master设备不发送新事务，然后将AINACTS置位。

10. 如果实现了ACE，则SoC在接口上发送了所有监听事务后，将输入引脚 ACINACTM 置位，使AXI master接口空闲。如果实施了CHI，则SoC将输入引脚 SINACT置位。当 L2 已完成 AXI master接口和salve接口的未完成事务时，STANDBYWFIL2 被置位以指示L2存储系统处于空闲状态。所有 Cortex-A53 处理器实现都包含 L2 存储系统，包括没有 L2 缓存的实现。

11. 当所有核心STANDBYWFI和STANDBYWFIL2都被置位时，集群准备好进入休眠模式。

12. 激活L2 cache RAM输入钳位。

13. 断开PDCPU和PDCORTEXA53 电源域的电源。

要退出休眠模式，请应用以下序列：

1. 应用正常的冷复位序列。必须对core和L2存储系统逻辑使用复位，直到电源恢复。在此复位序列期间，L2RSTDISABLE必须保持为高以禁用L2 cache硬件复位机制。

2. 电源恢复后，释放L2 cache RAM 输入钳位。

3. 在L2RSTDISABLE保持高电平的情况下继续正常的冷复位序列。

4. 如果需要，必须恢复架构状态。

#### Retention state

有关保留状态的信息，请联系 ARM。

### 使用WFE或SEV进行事件通信

外部代理可以使用 EVENTI 引脚参与与 Cortex-A53 处理器的 WFE 或 SEV 事件通信。当此引脚置位时，它会向设备中的所有core发送事件消息。这类似于在集群中的一个core上执行SEV指令。这使外部代理能够向core发出信号，表明它已释放信号量并且core可以离开WFE低功耗状态。EVENTI输入引脚必须保持高电平至少一个CLKIN时钟周期才能被core看到。

外部代理可以通过检查EVENTO引脚来确定集群中的至少一个core已经执行了**SEV指令**。当集群中的任何core执行 SEV 时，将向设备中的所有core发送一个事件信号，并且 EVENTO引脚被置位。当集群中的任何core执行SEV指令时，该引脚在三个CLKIN时钟周期内被置位为高电平。

### 与电源控制器的通信

Cortex-A53 处理器和系统电源管理控制器之间的通信可以使用以下之一或两者来执行：

• STANDBYWFI[3:0] 和STANDBYWFIL2 信号。

• Q-channel。

**STANDBYWFI[3:0] 和 STANDBYWFIL2 信号**

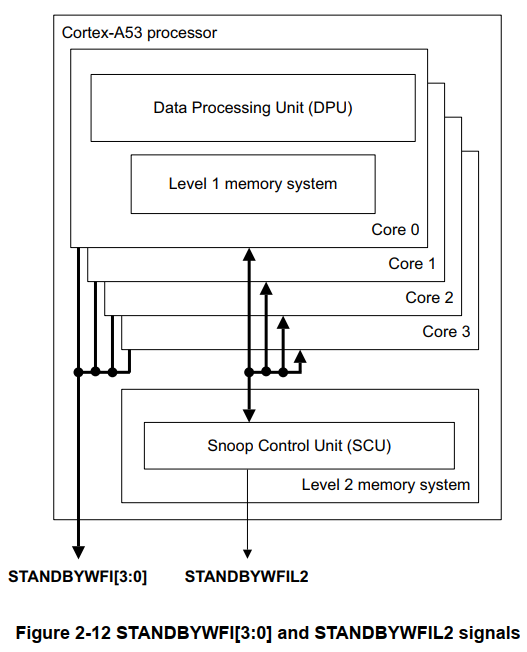
STANDBYWFI[n] 信号指示单个core何时处于空闲和低功耗状态。当STANDBYWFI[n] 被置位时，电源管理控制器可以从单个core中移除电源。有关详细信息，请参阅第 2-22 页的单个core关闭模式。

STANDBYWFIL2 信号指示所有单个core和 L2 存储系统何时处于空闲和低功耗状态。当 STANDBYWFIL2被置位时，电源管理控制器可以移除Cortex-A53 处理器的电源。有关详细信息，请参阅第 2-24 页的不带系统驱动的 L2 刷新的集群关闭模式和第 2-24 页的带系统驱动的 L2 刷新的集群关闭模式。

注意

Cortex-A53 处理器在没有 L2 cache的配置中包含最小的 L2 存储系统。因此，电源管理控制器必须始终等待 STANDBYWFIL2 的置位，然后才能从Cortex-A53处理器移除电源。

第 2-28 页的图 2-12 显示了 STANDBYWFI[3:0] 和 STANDBYWFIL2 如何对应各个core和 Cortex-A53 处理器。



**Q-channel**

Q通道是设备接口的时钟和电源控制器，用于管理设备的静止状态（quiescence）。该接口支持：

• 控制管理进入和退出设备静止状态。静止管理通常是，但不限于，设备或设备分区的时钟门控和电源门控保留状态。

• 指示退出静止状态。相关信号可以包含来自同一电源域中其他设备。

• 可选：拒绝静止请求。

• 跨时钟域的安全异步接口。

注意

有关详细信息，请参阅Low-Power Interface Specification: ARM® Q-Channel and P-Channel Interfaces。

# 编程模型

# 系统控制

## 关于

系统寄存器用来控制处理器实现的功能并提供状态信息。主要功能是：

• 整体系统控制和配置。

• MMU配置和管理。

• 缓存配置和管理。

• 系统性能监控。

• GIC配置和管理。

系统寄存器可在AArch64和AArch32执行态下访问。

在AArch64执行状态下访问的系统寄存器在AArch64寄存器描述中进行了描述。

在 AArch32 执行状态下访问的系统寄存器在AArch32寄存器描述中进行了描述。

一些系统寄存器可以通过内存映射或外部调试接口访问。

Armv7架构中描述的系统寄存器中的bits在Armv8-A架构中被重新定义：

• UNK/SBZP、RAZ/SBZP 和 RAZ/WI 被重新定义为 RES0。

• UNK/SBOP 和RAO/SBOP 被重新定义为RES1。

RES0和RES1在Arm®架构参考手册 Armv8 中进行了描述，用于Armv8-A架构配置文件。

有关执行状态的更多信息，请参阅 Arm® 架构参考手册 Armv8，了解 Armv8-A 架构配置文件。

### AArch32寄存器受CP15SDISABLE影响

在AArch32状态下，CP15SDISABLE输入禁用对某些系统寄存器的写访问。

Cortex-A53处理器没有任何受CP15SDISABLE影响的实现定义的寄存器。

有关受 CP15SDISABLE 影响的寄存器列表，请参阅 Arm® 架构参考手册 Armv8，了解 Armv8-A 架构配置文件。

（当 EL3 使用 AArch32 时，输入信号CP15SDISABLE为HIGH时禁用对某些安全寄存器的写访问。CP15SDISABLE不会影响以下方面：

• AArch64执行状态访问寄存器。

• 当EL3 使用AArch64 并且EL1使用AArch32时，来自安全EL1的访问寄存器。）

### AArch64 register summary

### AArch64 register descriptions

### AArch32 register summary

### AArch32 register descriptions

# MMU

Memory Management Unit

## 关于

Cortex-A53 处理器是兼容 Armv8 的处理器，支持在 AArch64 和 AArch32 状态下执行。在 AArch32 状态下，Armv8 地址转换系统类似于带有 LPAE 和虚拟化扩展的 Armv7 地址转换系统。在 AArch64 状态下，Armv8 地址转换系统类似于对长描述符格式地址转换系统的扩展，以支持扩展的虚拟和物理地址空间。有关地址转换格式的更多信息，请参阅 Arm® 架构参考手册 Armv8，了解 Armv8-A 架构配置文件。 AArch64 和 AArch32 地址转换系统之间的主要区别在于 AArch64 状态提供：

• 4KB 或 64KB 的转换粒度。在 AArch32 中，转换粒度被限制为 4KB。

• 16位ASID。在 AArch32 中，ASID 被限制为8位值。

支持的最大物理地址大小为40 位。

您可以单独启用或禁用地址转换的每个阶段。

MMU 控制访问主存储器中的转换表的表遍历硬件。 MMU 将虚拟地址转换为物理地址。 MMU 通过一组虚拟到物理地址映射和页表中保存的存储属性提供细粒度的存储系统控制。当访问某个位置时，它们会被加载到翻译后备缓冲区 (TLB，Translation Lookaside Buffer) 中。

每个core中的 MMU 具有以下特点：

• 10-entry fully-associative instruction micro TLB.

• 10-entry fully-associative data micro TLB.

• 4-way set-associative 512-entry unified main TLB.

• 4-way set-associative 64-entry walk cache.

• 4-way set-associative 64-entry IPA cache.

• TLB 条目包括全局和应用程序特定标识符，以防止上下文切换 TLB 刷新。

• 虚拟机标识符 (VMID)，用于防止虚拟机管理程序在虚拟机切换时刷新TLB。

## TLB organization

## TLB match process

## External aborts

# L1存储系统

## 关于

L1 存储系统由独立的指令和数据cache组成。实现者在实现过程中独立配置指令和数据缓存，大小分别为 8KB、16KB、32KB 或 64KB。

L1 指令存储系统具有以下主要特性：

• 指令cache line长度为 64 byte。

• 2-way set associative L1 Instruction cache。

• L2 存储系统的128-bit读接口。

L1 数据存储系统具有以下特性：

• 数据cache line长度为 64 bytes。

• 4-way set associative L1 data cache。

• L2 存储系统的256-bit写接口。

• L2 存储系统的128-bit读接口。

• 为数据高速缓存单元（DCU, Data Cache Unit）、指令提取单元（IFU, Instruction Fetch Unit）和TLB 服务的读缓冲区。

• 从数据 L1 存储系统到数据路径的 64-bit读取路径。

• 从数据路径到L1 存储系统的128-bit写路径。

• 支持三个未完成的数据缓存未命中。

• 合并store缓冲区功能。这处理写：

— Device memory.

— Normal Cacheable memory.

— Normal Non-cacheable memory.

• 数据端预取引擎。

## Cache behavior

## Support for v8 memory types

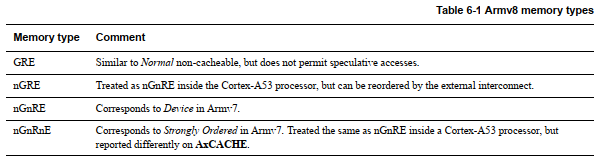
Armv8-A 架构引入了几种新的存储类型来代替 Armv7 Devic和Strongly-Ordered存储类型。这些与以下属性有关：

G Gathering。将请求收集并合并到单个事务中的能力。

R Reordering。重新排序传输的能力。

E Early-acknowleadge。接受来自互连的事务的早期确认的能力。

表 6-1 描述了 Armv8 内存类型。



根据架构的定义，这些位仅在转换表被标记为Armv8 Device memory时适用，它们不适用于Normal memory。如果 Armv7 架构操作系统在 Cortex-A53 处理器上运行，则 Device memory类型匹配nGnRE编码，Strongly-Ordered memory类型匹配 nGnRnE memory类型。

## L1 Instruction memory system

## L1 Data memory system

## Data prefetching

## Direct access to internal memory

# L2存储系统

## 关于

L2 存储系统包括：

• Snoop Control Unit(SCU)，最多可连接集群内的四个core。 SCU 包含L1 data cache tag的副本，用于维护一致性。L2 存储器系统通过 AMBA 4 ACE 总线或 AMBA 5 CHI 总线与外部存储系统连接。所有总线接口均为 128 位宽。

• 可选配紧密耦合的L2 Cache，包括：

— 可配置的二级缓存大小为 128KB、256KB、512KB、1MB 和 2MB。

— 固定line长度为 64 字节。

— 物理索引和标记缓存。

— 16 路组关联缓存结构。

— 如果配置了L2缓存，则可选配ACP 接口。

— 可选的 ECC 保护。

L2 存储系统具有同步中止机制和异步中止机制，请参阅第 7-18 页的外部中止处理。

## SCU

SCU，Snoop Control Unit

Cortex-A53处理器支持1到4个由SCU维护L1数据缓存一致性的独立core。 SCU与core同频。

SCU使用与MOESI状态等效的ACE状态来维护一致性。

SCU包含的缓冲区可以处理core之间的cache-to-cache传输，不必访问访问外部存储系统。Cache line的dirty副本可以在core间移动，且不需要写回外部存储系统。

每个core都包含tag和dirty RAMs标记cacheline的状态。 SCU也包含副本tag，在进行snoop时，不必为每个一致性请求都去snoop 该cluster的其他core。副本tag可以保证core和系统的一致性请求高效运行。

当外部snoop在副本tag中命中时，会向对应的core发出请求。

### 总线接口配置信号

Cortex-A53包含以心接口配置信号：

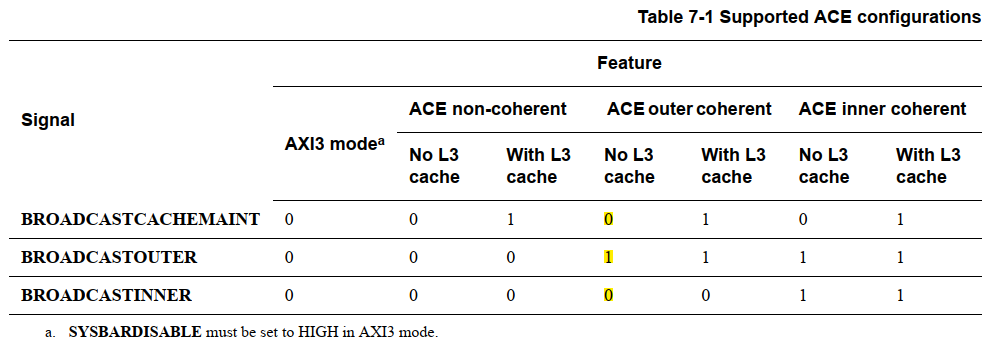
• BROADCASTINNER.

• BROADCASTOUTER.

• BROADCASTCACHEMAINT.

• SYSBARDISABLE (ACE only).

表7-1显示了配置为ACE总线时，这些信号的允许组合以及Cortex-A53处理器中支持的配置



下图显示了ACE配置下的主要特性

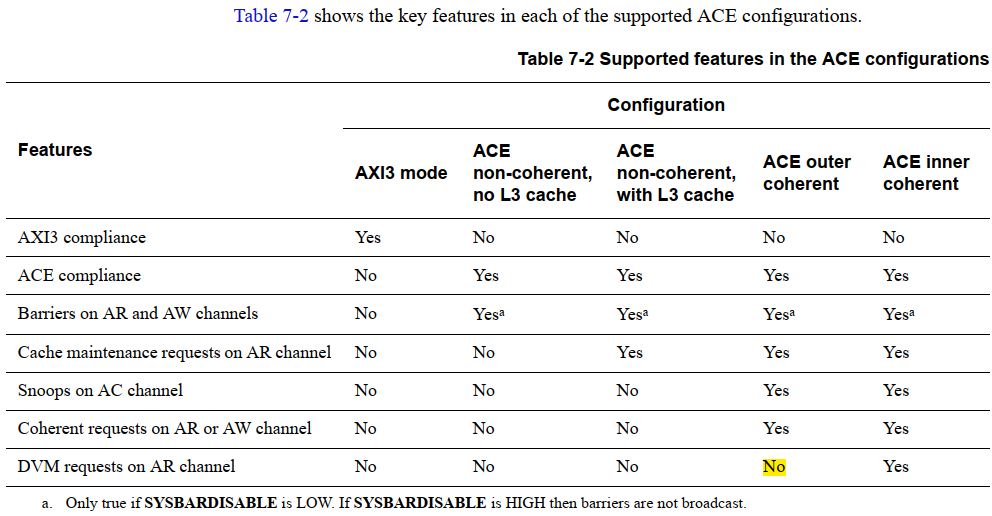
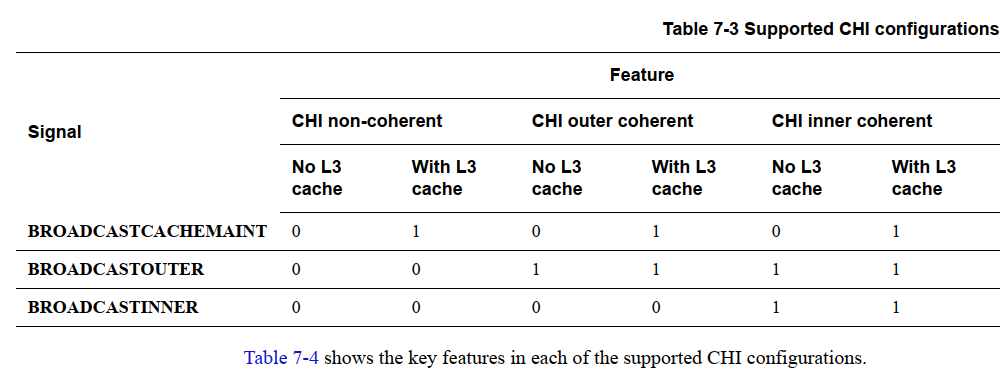
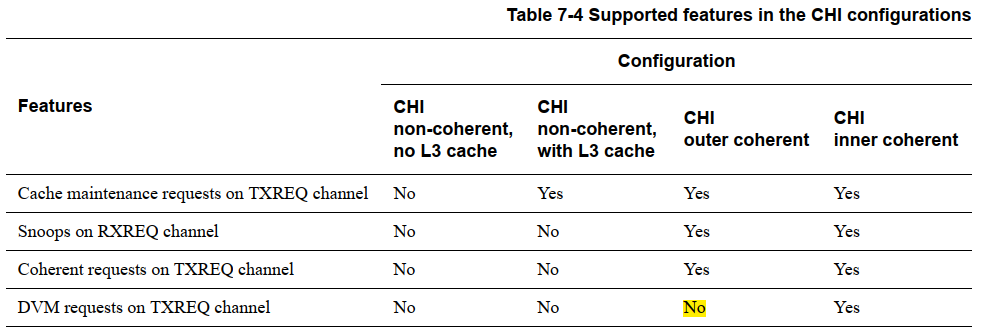


表7-3显示了配置为CHI总线时，这些信号的允许组合以及Cortex-A53处理器中支持的配置



下图显示了CHI配置下的主要特性



### 监听和维护请求

Snoop and maintenance requests

SCU使用外部BROADCASTINNER、BROADCASTOUTER 和 BROADCASTCACHEMAINT 引脚控制对系统的探听和维护请求：

• 当将BROADCASTINNER引脚设置为1时，inner shareability domain扩展到 Cortex-A53 处理器之外，并且Inner Shareable snoop and maintenance操作被广播外部。当将 BROADCASTINNER 引脚设置为0时，内部共享域不会超出 Cortex-A53 处理器。

• 当将BROADCASTOUTER引脚设置为1时，外部共享域会扩展到Cortex-A53 处理器之外，并且外部可共享监听和维护操作会在外部广播。当将BROADCASTOUTER引脚设置为0时，外部共享域不会超出Cortex-A53处理器。

• 当将BROADCASTCACHEMAINT引脚设置为1时，这向 Cortex-A53 处理器表明存在外部下游Cache并且维护操作需要在外部广播。当将BROADCASTCACHEMAINT引脚设置为0时，Cortex-A53 处理器外部没有下游缓存。

注意

• 如果您将 BROADCASTINNER 引脚设置为 1，还必须将BROADCASTOUTER引脚设置为 1。

• 在包含Cortex-A53处理器和big.LITTLE™配置中的其他处理器的系统中，必须确保 BROADCASTINNER和BROADCASTOUTER引脚打开两个处理器都设置为HIGH，以便两个处理器位于同一个内部共享域中。

• Cacheable load和store在不超出Cortex-A53处理器的共享域，可以将数据allocate给L1和L2缓存。但是，它们不会针对这些访问在master口上发出一致的请求，只使用 ReadNoSnoop 或 WriteNoSnoop 事务。包括non-shareable memory空间，且可能包括inner shareable和outer shareable memory，具体取决于 BROADCASTINNER 和 BROADCASTOUTER 引脚的设置。

• 如果系统向 Cortex-A53 处理器发送snoop以获取L1或L2缓存中存在的地址，但缓存中的line属于不超出cluster的共享域中，则该snoop以miss作为响应对待。

【使用说明】

1、如果外部总线有snoop filter（满足cache一致性、I/O一致性应用需求），或无snoop filter但多Cluster有cache一致性需求，BROADCASTOUTER应该配置位1。因为配0的话，发出的操作是No-snoop类型，状态在snoop filter里不记录，也无法产生对其他Cluster的监听请求。

2、如果外部总线无snoop filter且只有一个Cluster （无cache一致性需求），BROADCASTOUTER可以配置为0，A53发出的操作都是No-Snoop类型。关注需要I/O一致性的总线方案，在收到从HAC发出的Snoop类型的命令后，是否会产生对A53的snoop操作？

3、

## ACE master

本节介绍 ACE master接口的属性。ACE 接口可以以 CLKIN 频率的整数比进行门控。

### 存储器接口属性

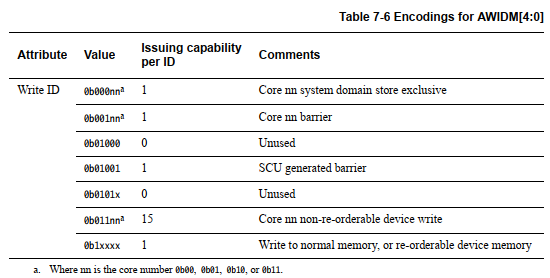
表 7-5 显示了 Cortex-A53 处理器的ACE master接口属性。如果处理器包含四个core，该表列出了读和写发射能力的最大可能值。

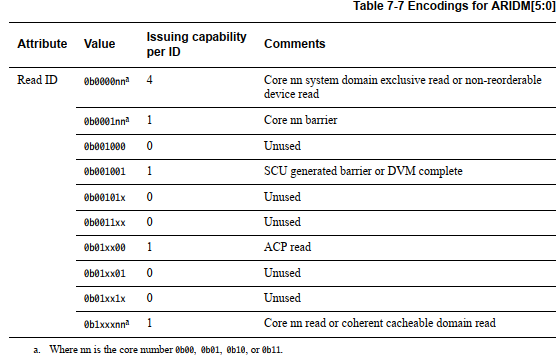
表 7-5 ACE master接口属性

|  |  |  |
| --- | --- | --- |
| Attribute | Valuea | Comments |
| 写发射能力 | 17 + n | 集群最多可以发出16个写，不包括barrier：  • 最多16个写Normal memory，inner和outers都为write-back cacheable的区域；  • 最多15个写其它memory类型，包括Device、Normal non-cacheable和Write-through。  可能有其他混合memory类型，并且写可能是Single写也有可能是Burst写。  每个core也可以发出一个barrier，集群可以发出一个额外的barrier。 |
| 读发射能力 | 8n + 4m + 1 | 集群中每个core的8包括最多：  • 8 data linefills.  • 4 non-cacheable or Device data reads.  • 1 non-cacheable TLB page-walk read.  • 3 instruction linefills.  • 5 coherency operations.  • 1 barrier operation.  • 8 DVM messages  注意  每个core的8条DVM消息都可以是两部分的DVM消息，从而导致每个core最多16个DVM事务​​。  如果配置了ACP，则最多可以生成4个ACP linefill 请求。  可以从集群中生成 1 个barrier操作。 |
| 排他访问 | n | 每个core可以有1个排他访问正在进行中。 |
| 写 ID能力 | 17 + n | 未完成写ID的最大数量为21。这与未完成写的最大数量相同。  只有具有nGnRnE或nGnRE的Device memory类型才能具有多个相同AXI ID的未完成的事务。所有其他memory类型类型的未完成的事务使用各自唯一的AXI ID。 |
| 写ID位宽 | 5 | ID 对memory事务的源进行编码。见表 7-6。 |
| 读ID能力 | 8n + 4m + 1 | 集群中每个核心的 8 个，此外还有：  • 4 个用于 ACP。  • 1 表示barrier。  只有具有 nGnRnE 或 nGnRE 的Device memory类型才能具有多个相同AXI ID的未完成的事务。所有其他memory类型的未完成事务使用各自唯一的AXI ID。  两部分的DVM使用相同的ID，因此可以在同一ID上有两个未完成的事务。 |
| 读ID位宽 | 6 | ID 对memory事务的源进行编码。请参见表 7-7。 |

a. n is the number of cores.

m is 1 if the processor is configured with an ACP interface, and 0 otherwise.





注意

这些ID和事务细节仅供参考。 Arm 强烈建议所有互连和外设都设计为，支持任何ID上的任何类型和数量的事务，以确保与未来产品的兼容性。（不根据ID信息做特殊设计）

### ACE传输

Cortex-A53 处理器不会生成任何FIXED突发，并且所有WRAP 突发都会首先从关键字开始获取完整的cache line。突发不跨越cache line边界。

Cache line填充获取长度始终为64 byte。

Cortex-A53 处理器仅在master接口上生成所支持的AXI事务的一个子集。

对于 WriteBack 传输，支持的传输是：

• WRAP 4 128 bit读传输（linefills）。

• INCR 4 128 bit写传输（evictions）。

• INCR N（N:1、2 或 4）128 bit写传输（read allocate）。

对于Non-cacheable的事务：

• INCR N（N:1、2 或 4）128 bit写传输。

• INCR N（N:1、2 或 4）128 bit读传输。

• INCR 1 读传输，支持：32bit、64bit和128bit。

• INCR 1 写传输，支持：8bit、16bit、32bit、64bit和128bit。

• INCR 1 exclusive写，支持：8bit、16bit、32bit、64bit和128bit。

• INCR 1 exclusive读，支持：8bit、16bit、32bit、64bit和128bit。

对于Device事务：

• INCR N（N:1、2 或 4）128 bit读传输。

• INCR N（N:1、2 或 4）128 bit写传输。

• INCR 1 读传输，支持：8bit、16bit、32bit、64bit和128bit。

• INCR 1 写传输，支持：8bit、16bit、32bit、64bit和128bit。

• INCR 1 exclusive读，支持：8bit、16bit、32bit、64bit和128bit。

• INCR 1 exclusive写，支持：8bit、16bit、32bit、64bit和128bit。

对于translation table walk事务：

INCR 1 32 bit和 64 bit读传输。

以下几点适用于 AXI 事务：

• WRAP 突发只有 128 bit。

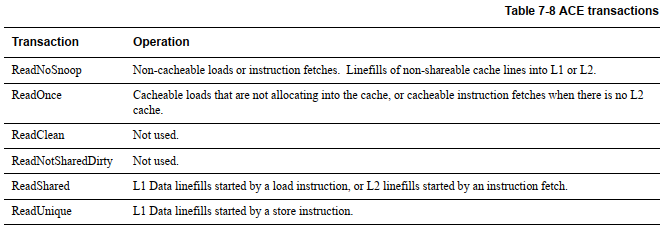
• INCR 1 可以是任意size的读或写。

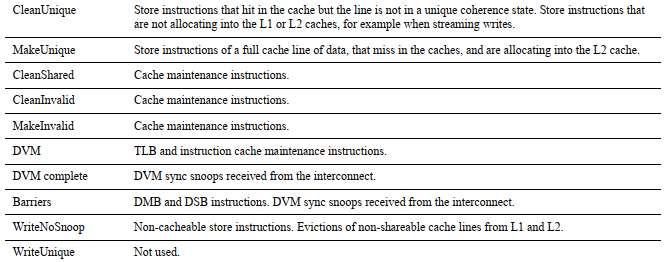
• INCR 突发长度多于一的，只有128位。

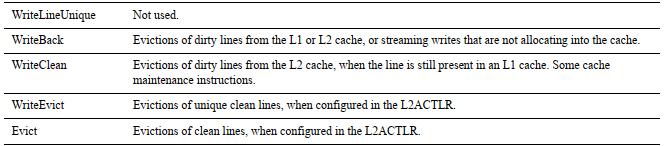
• 没有FIXED传输。

• 写传输的strob可能全有效、部分有效或全无效。

表 7-8 显示了可以生成的 ACE 事务，以及可能导致生成事务的一些典型操作。这不是生成每种类型传输的场景的详尽列表，因为有很多可能性。







### Snoop通道属性

表 7-9 ACE 通道属性

|  |  |  |
| --- | --- | --- |
| Property | Valuea | Comments |
| Snoop接受能力 | 8 | SCU 最多可以接受和处理来自系统的 8 个 snoop 请求。它计算从AC通道上接受的请求到CR通道上接受的响应的请求。 |
| Snoop latency | Hit | 当 L2 cache命中时，响应和数据的最佳情况是 13 个处理器周期。当 L2 cache未命中而 L1 cache命中时，响应和数据的最佳情况是 16 个处理器周期。  注意  如果发生危险(hazards)或没有足够的缓冲区来吸收请求，则延迟可能会更高。 |
| Miss | 当SCU 副本tag和L2 tag指示未命中时，最好的情况是6个处理器周期。 |
| DVM | 集群至少需要6个周期来响应 DVM 数据包。 |
| Snoop filter | Supported | 集群为互连中的外部监听过滤器提供支持。它通过在写通道上发送Evict事务来指示何时从处理器中逐出干净的行。但是，在某些情况下，不正确的软件可能会阻止发送 Evict 传输。因此，您必须确保构建的外部监听过滤器能处理容量溢出，如果存储空间用完，该容量溢出会向处理器发送back-invalidation。  不产生Evict的例子包括：  • Linefills that take external aborts.  • Store exclusives that fail.  • Mis-matched aliases |
| 支持的事务 | - | ACE 协议描述的所有事务：  • 在系统的主接口上接受。  • 可以在ACE 主接口上生成，除了：  — WriteUnique。  — WriteLineUnique。  — ReadNotSharedDirty。  — ReadClean。 |

### 读响应

ACE master设备可以通过将RREADY保持为低电平来延迟接受读数据通道传输一个不确定的周期数。在同一事务的部分读数据通道传输之间，可以将 RREADY 置为低电平。

在接受事务的最后一个读数据通道传输后，ACE master在ACLK周期后拉高读确认信号 RACK。除了ACE配置，RACK在 AXI3 兼容模式下被置位。

注意

• 为了系统组件的互操作性，Arm建议与ACE master接口的组件完全符合ACE标准，而不依赖于为Cortex-A53处理器描述的允许RACK行为的子集。

• 如果互连总线不支持一致性和非一致性请求之间的冒险(hazarding)，则在它为非一致性读返回第一次读数据传输后，它必须返回事务中所有剩余的读传输，不需要对集群发送相同的地址snoop。

### 写响应

ACE master要求slave在收到写地址之前不返回写响应。

ACE master始终通过将BREADY保持为高电平来立即接受写响应。

ACE master在接受写响应后的ACLK周期中将写确认信号WACK置为HIGH有效。除了 ACE配置，WACK在AXI3兼容模式下被置位。

注意

为了系统组件的互操作性，Arm 建议与ACE master接口的组件完全符合 ACE 标准，而不依赖于为Cortex-A53处理器描述的允许的BREADY和WACK行为的子集。

### Barriers

Cortex-A53 处理器支持向互连发送barrier事务，或终止集群内的barrier。

• 要在ACE接口上发送barrier，请将 SYSBARDISABLE 设置为 LOW。

• 要终止集群内的barrier，请将SYSBARDISABLE 设置为HIGH。

如果您终止集群内的barrier，请确保您的互连和连接到它的任何外设不会返回事务的写响应，直到该事务被后面的barrier视为完成。这意味着写必须对系统中的所有其他master都是可观察的。 Arm 预计大多数外设都能满足这一要求。

为了获得最佳性能，Arm 建议在集群内终止Barrier。

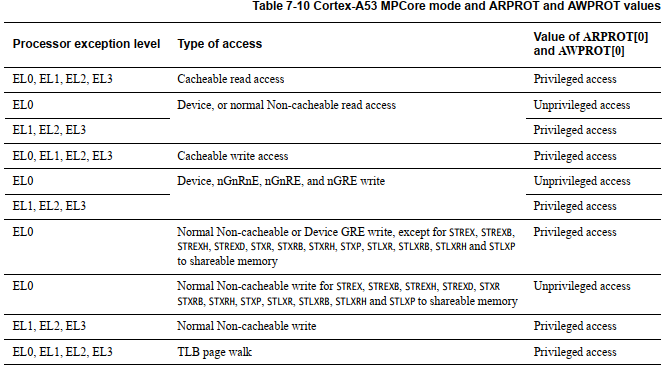
### AXI3 兼容模式

Cortex-A53 处理器实现了AXI3兼容模式，使您能够在不需要AMBA 4 ACE 接口且处理器不会在集群外传播Barrier的独立环境中使用处理器。要启用此模式，您必须在处理器边界将 SYSBARDISABLE 输入引脚设置为HIGH。您还必须确保将 BROADCASTINNER、BROADCASTOUTER 和 BROADCASTCACHEMAINT 输入引脚设置为 LOW。

### AXI 特权信息

AXI 使用ARPROTM[0] 和 AWPROTM[0] 信号提供了有关访问的特权级别的信息。但是，当访问可能被缓存或合并在一起时，生成的事务可以将特权数据和非特权数据组合在一起。如果发生这种情况，Cortex-A53 处理器会将事务标记为特权，即使它是由非特权进程发起的。

表 7-10 显示了 Cortex-A53 处理器异常级别和相应的 ARPROTM[0] 和 AWPROTM[0] 值。



## CHI master

NA

## 附加的memory属性

Cortex-A53 处理器通过降级某些memory类型来简化一致性逻辑：

• 标记为Inner Write-Back Cacheable和Outer Write-Back Cacheable的memory会缓存在 L1 Data cache和 L2 cache中。

• 标记为Inner Write-Through 的memory被降级为Non-cacheable。

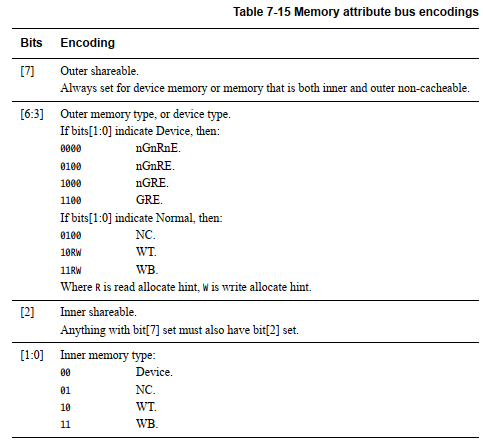
• 标记为Outer Wite-Through或Outer Non-cacheable的memory将降级为Non-cacheable，即使inner属性是Write-Back cacheable的。

ACE的ARCACHE或AWCACHE 或 CHI 的MemAttr和SnpAttr上提供的属性是这些降级属性，并指示互连必须如何处理事务。

某些互连或总线协议可能需要有关memory类型的更多信息，对于这些情况，集群会从存储在TLB中的转换表中导出未更改的memory属性信息。这些信号仅供参考，不构成ACE 或CHI协议的一部分。

在ACE的配置下，有一个用于读通道的RDMEMATTR 总线和一个用于写通道的 WRMEMATTR 总线。

在 CHI的配置下，只有一条REQMEMATTR 总线。表 7-15 描述了内存属性总线上的编码。



## 可选集成L2 Cache

可选集成L2可配置cache大小为128KB、256KB、512KB、1MB 和 2MB。

**数据**仅在从L1存储系统逐出时才分配到L2 cache，而不是在首次从系统中获取时分配。此规则的唯一例外是标记为inner transient hint的memory，或non-temporal loads，请参阅第 6-12 页的Non-temporal loads，它们只在L2 分配缓存。 L1 cache可以从系统中预取数据，而不必从L2 cache中逐出数据。

**指令**在从系统中获取时分配到L2 cache，并且可以在maintenance操作期间无效掉。

L2 cache是16-way set associative。 L2 cache tags与SCU副本tag并行查找。如果 L2 tag和 SCU 副本tag都命中，则读会优先访问L2 cache，而不是snoop其他core。

L2 RAM在复位时自动无效化，除非在nL2RESET信号无效时L2RSTDISABLE信号设置为高电平。

### 外部中止处理

External aborts handling

存储器系统使用同步中止机制、异步中止机制或nEXTERRIRQ引脚处理外部中止，如下所示：

**同步中止机制 Synchronous abort mechanisms**

以下访问的外部中止使用同步中止机制：

• 所有load访问。

• 所有store exclusive访问（STREX、STREXB、STREXH、STREXD、STXR、STXRB、STXRH、STXP、STLXR、STLXRB、STLXRH 和 STLXP）。

**异步中止机制 Asyncronous abort mechanisms**

以下访问的外部中止使用异步中止机制：

• Store到Device memory（Store Exclusive访问除外）。

• Store到Normal memory，且属性为Inner Non-cacheable, Inner Write-Through, Outer Non-cacheable, Outer Write-Through （Store exclusive访问除外）。

• 从互连总线收到ditry状态的返回的数据填充L1 Data cache和L2 cache。

**nEXTERRIRQ 引脚**

下述访问的外部中止，导致 nEXTERRIRQ引脚被置位，因为中止可能不直接与集群中的特定core相关。

• 对Normal memory的所有store访问，且属性为Inner Write-Back 和 Outer Wirte-Back.

• 从L1 data cache或 L2 cache中逐出。

• DVM 完成交易。

注意

当 nEXTERRIRQ 被置位时，它会一直保持置位，直到通过向L2ECTLR寄存器的AXI或CHI异步错误位写0来清除错误。

## ACP

可选的加速器一致性端口 (ACP，Accelerator Coherency Port) 作为AXI4 slave接口实现，但具有以下限制：

• 128 位读写接口。

• ARCACHE 和AWCACHE 仅限于Normal、Write-Back、Read-Write-Allocate、Read-Allocate、Wire-Allocate和No-Allocate。 ARCACHE 和 AWCACHE 仅限于值 0b0111、0b1011 和 0b1111。其他值会导致RRESP或BRESP上的SLVERR响应。

• 不支持排他访问。

• 不支持Barrier。写事务的BRESP表明该写的全局可观察性。

• ARSIZE 和AWSIZE 信号不存在，假定值为0b100，16 byte。

• ARBURST 和AWBURST 信号不存在，取值INCR。

• ARLOCK 和AWLOCK 信号不存在。

• ARQOS 和AWQOS 信号不存在。

• ARLEN 和AWLEN 的值限制为0 和3。

本节介绍ACP：

• 传输大小支持。

• ACP user信号。

• ACP 性能。

### 传输大小

ACP支持以下读请求传输大小和长度组合：

• 64 字节 INCR 请求的特征是：

— ARLEN 为 0x03，4拍。

— ARADDR 与 64 字节边界对齐，因此 ARADDR[5:0] 为 0b000000。

— ARSIZE 和 ARBURST 分别假定值为0b100和INCR。

• 16 字节INCR 请求的特征是：

— ARLEN 为 0x00，1 拍。

— ARADDR 与 16 字节边界对齐，因此 ARADDR[3:0] 为 0x0。

ACP 支持以下写请求传输大小和长度组合：

• 64 字节 INCR 请求的特征是：

— AWLEN 为 0x03，4拍。

— AWADDR 与 64 字节边界对齐，因此 AWADDR[5:0] 为 0b000000。

— AWSIZE 和 AWBURST 分别假定值为 0b100 和 INCR。

— 所有节拍的WSTRB必须相同，并且全部有效或全部无效。

• 16 字节INCR 请求的特征是：

— AWLEN 为0x00，1 拍。

— AWADDR 与 16 字节边界对齐，因此 AWADDR[3:0] 为 0x0。

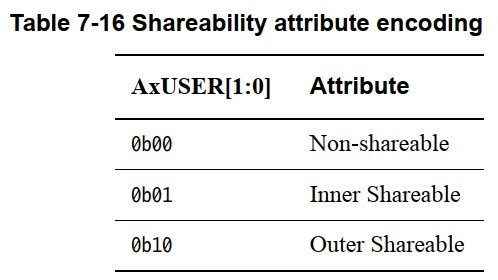
— AWSIZE 和 AWBURST 分别假定值为 0b100 和 INCR。

— WSTRB 可以取任何值。

不满足AR和AW通道限制的请求会导致 RRESP 或 BRESP 上的 SLVERR 响应。

### ACP user信号

ACP事务可以引发对系统的一致请求。因此ACP请求必须将Inner和Outer Shareable 属性传递给L2。要传递可共享性属性，请使用表 7-16 中描述的编码。有关详细信息，请参阅第 A-23 页的 ACP 接口信号。



注意

这与ACE上的AxDOMAIN编码相同，但不支持值0b11。

### ACP性能

ACP 接口最多可以支持4 outstanding事务。可以是读和写的任意组合。

master必须避免使用相同AXI ID上连续发送多个未完成的事务，防止第二个事务在第一个事务完成之前暂停。如果master需要在两个事务之间保序，Arm建议在发送第二个事务之前等待对第一个事务的响应。

当写包含完整的cacheline数据时，写通常具有更高的性能。

如果配置了SCU cache保护，则小于64 bit的写会在命中L2 cache时产生额外的读-改-写序列的开销。

一些 L2 资源在 ACP 接口和core之间共享，因此ACP接口上的大量流量在某些情况下可能会降低core的性能。

您可以使用ARCACHE和AWCACHE信号来控制ACP请求是否会在未命中时导致分配到 L2 cache中。但是，如果配置了CHI master接口，则为了确保数据拍的正确顺序，ACP读取的未命中总是分配到L2 cache中。

# Cache保护

## cache保护行为

Cortex-A53 处理器可防止软错误导致RAM bitcel暂时保持不正确的值。处理器将新值写入RAM以纠正错误。如果错误是硬错误，不能通过写RAM来纠正，例如RAM中的物理缺陷，那么处理器可能会在不断检测并尝试纠正错误时进入活锁(livelock)。

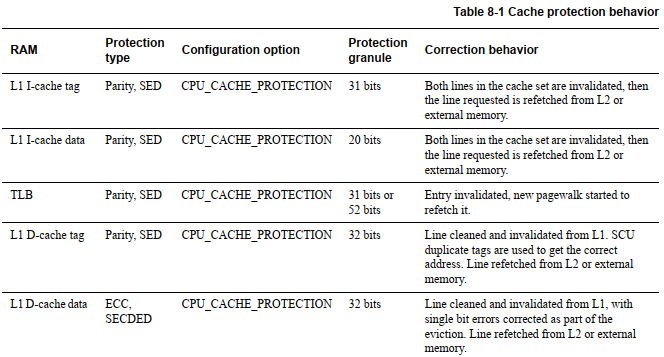
一些RAM具有单错误检测 (SED, Single Error Detect) 功能，而其他RAM具有单错误纠正、双错误检测 (SECDED，Single Error Correct, Double Error Detect) 功能。 L1 data cache dirty RAM 是单错误检测、单错误纠正 (SEDSEC, Single Error Detect, Single Error Correct)。当任何 RAM 中存在单个位错误时，处理器可继续执行并保持功能正确。如果在不同的 RAM 中或在同一 RAM 的不同保护颗粒中存在多个单比特错误，则处理器在功能上也保持正确。如果同一保护颗粒内的单个RAM中存在双位错误，则行为取决于 RAM：

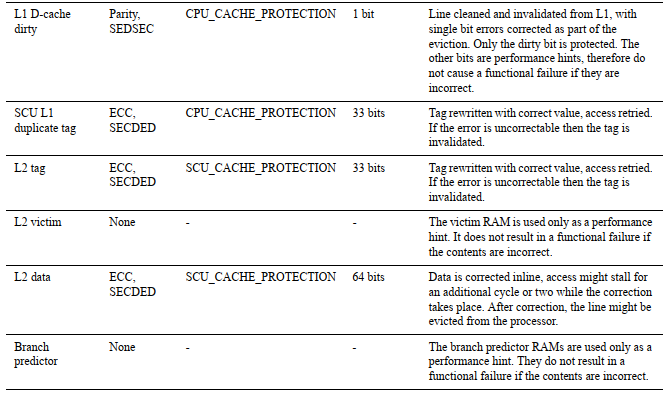
• 对于表 8-1 中列出的具有SECDED 功能的 RAM，将按照错误报告中所述检测并报告错误第 8-4 页。如果错误出现在包含脏数据的高速缓存行中，则该数据可能会丢失，从而导致数据损坏。

• 对于只有SED的RAM，不会检测到双位错误，因此可能会导致数据损坏。

如果存在三个或更多位错误，则取决于RAM和RAM中错误的位置，可能会检测到错误，也可能不会检测到错误。

当不存在错误时，Cortex-A53 CPU 缓存保护支持对性能的影响最小。当检测到错误时，导致错误的访问在进行更正时停止。当更正完成时，访问或者继续使用更正的数据，或者重试。如果访问被重试，它要么用更正的数据再次命中高速缓存，要么在高速缓存中未命中并从较低级别的高速缓存或主存储器重新获取数据。每个RAM的行为如表 8-1 所示。





注意

如果在需要多个周期才能完成的load指令（例如 LDM）的第一次data cache访​​问之后,发生可纠正的ECC错误，并且发生以下情况之一：

• 已设置硬件断点、观察点或向量捕获自重新执行时触发的第一次执行以来。

• 自第一次执行以来页表已被修改，导致重新执行时出现指令或数据中止陷阱。

在发生可纠正的 ECC 错误之前，使用已成功读取的数据更新寄存器文件。

## Error报告

CPUMERRSR 或 L2MERRSR 寄存器中报告检测到的任何错误。请参见第 4-120 页的 CPU Memory Error Syndrome Register或第 4-123 页的L2 Memory Error Syndrome Register。检测到的任何错误也会在PMUEVENT总线上发出信号。请参阅第 12-35 页的Events。这包括成功纠正的错误和无法纠正的错误。如果在同一时钟周期内发生多个错误，则只报告其中一个。

无法纠正并因此可能导致数据损坏的错误也会导致中止或外部引脚被置位，以便软件可以意识到存在错误并可以尝试恢复或重新启动系统：

• 无法纠正L2 data RAM 在被取指、TLB pagewalk 或load指令读取时出现错误，可能会导致精确数据中止或预取中止。

• 当load、store或预加载指令或硬件预取器读取到L1 data cache时，L2 data RAM 中的不可纠正错误可能会导致异步异常。

• 当从cache中逐出行时，L1 或L2 data RAM 中的不可纠正错误导致nINTERRIRQ 引脚被置位。这可能是由于自动逐出、缓存维护操作或snoop。

• L2 tag RAM 或 SCU L1 副本tag RAM 中不可纠正的错误导致 nINTERRIRQ 引脚被置位。

注意

• 当nINTERRIRQ 被置位时，它一直保持置位，直到通过向L2ECTLR 寄存器的L2 内部异步错误位写0 来清除错误。

• Arm 建议将 nINTERRIRQ 管脚连接到中断控制器，以便在管脚被置位时产生中断或系统错误。

当data RAM 上有错误的dirty cacheline从处理器中逐出时，master接口上的写仍然发生，但是如果错误无法纠正，则：

• 在 ACE 上，写strobe置无效，因此不正确数据不写入外部。

• 在 CHI 上，strobe置有效，但响应字段表明存在数据错误。

当snoop命中具有不可纠正数据错误的cache line时，如果snoop需要，则返回数据，但snoop响应指示存在错误。

如果snoop命中具有不可纠正错误的tag，则将其视为snoop未命中，因为该错误意味着不知道cache line是否有效。

注意

在某些情况下，可能会多次计算错误。例如，作为纠正过程的一部分，在行被逐出之前，多次访问可能会读取有错误的位置。

## Error注入

Error injection

为了支持错误处理软件的测试，Cortex-A53 处理器提供了强制将双位错误注入L1 D-cache data RAM、L2 data RAM 和 L2 tag RAM 的能力。

通过设置 CPUACTLR.L1DEIEN 位启用 L1 D-cache data RAM 上的错误注入。当该位被设置时，双位错误被注入到对每个32-byte区域的第一个字的 L1 D-cache data RAM 的所有写操作中。这对应于位 [4:2] 为 0b000 的地址的字节。可以写入 L1 D-cache RAM 的原因是：

• 来自core的显式store。

• 由于以下原因，cache 取cache line：

— Load instructions.

— Store instructions.

— Preload instructions.

— Data prefetches.

— Pagewalks

通过设置L2ACTLR.L2DEIEN 位来启用 L2 data RAM上的错误注入。当该位被设置时，双位错误被注入到 L2 cache data RAM 的所有写操作中。可以写入 L2 data RAM 的原因是：

• Explicit stores from one of the cores.

• Instruction fetches or prefetches.

• Evictions from the L1 Data cache.

• ACP accesses.

通过设置 L2ACTLR.L2TEIEN 位来启用 L2 tag RAM 上的错误注入。当该位被设置时，双位错误被注入到 L2 tag RAM 的所有写操作中。可以写入 L2 cache tag RAM 的原因是：

• Explicit stores from one of the cores.

• L2 allocations caused by instruction fetches or prefetches.

• Evictions from the L1 Data cache.

• ACP accesses.

• Snoop operations.

• Cache maintenance instructions.

# GIC CPU接口

## 关于

GIC CPU 接口与外部distributor组件集成时，是用于支持和管理集群系统中的中断的资源。它提供：

• 用于管理的寄存器：

— 中断源。

— 中断行为。

— 中断路由到一个或多个core。

Cortex-A53 处理器实现GIC CPU接口，如通用中断控制器 (GICv4) 架构中所述。它与系统内的外部GICv3或GICv4中断分发器组件连接。

GICv4 架构支持：

• 两种安全状态。

• 中断虚拟化。

• 软件产生的中断(SGI)。

• 基于消息的中断。 (ITS，不配置)

• 系统寄存器访问。

• 存储映射寄存器访问。

• 中断屏蔽和优先级。

• 集群环境，包括包含八个以上core的系统。

• 电源管理环境中的唤醒事件。

GIC 包括中断分组功能，支持：

• 根据软件配置，使用IRQ 或FIQ 异常请求向目标core发送中断组信号。

• 处理Group 0 和Group 1 中断优先级的统一方案。

本章仅描述特定于 Cortex-A53 处理器实现的功能。

### 旁路CPU接口

GIC CPU 接口始终在 Cortex-A53 处理器中实现。但是，如果您在复位时将 GICCDISABLE 信号置为高电平，则可以禁用它。如果启用 GIC，输入引脚 nVIRQ 和 nVFIQ 必须连接到高电平。这是因为内部 GIC CPU 接口会向core生成虚拟中断信号。 nIRQ 和 nFIQ 信号由软件控制，因此无需将它们连接为高电平。如果禁用 GIC CPU 接口，输入引脚 nVIRQ 和 nVFIQ 可以由SoC中的外部 GIC 驱动。

当Cortex-A53处理器未与系统中的外部GICv3或GICv4分配器组件集成时，通过在复位时将 GICCDISABLE 信号置为高电平来禁用CPU接口。

在复位时将 GICCDISABLE 信号置为高电平会移除对存储映射和系统GIC CPU接口寄存器的访问。

## GIC编程模型

# Generic Timer

## 关于

通用定时器可以根据递增的计数器值安排事件和触发中断。它提供：

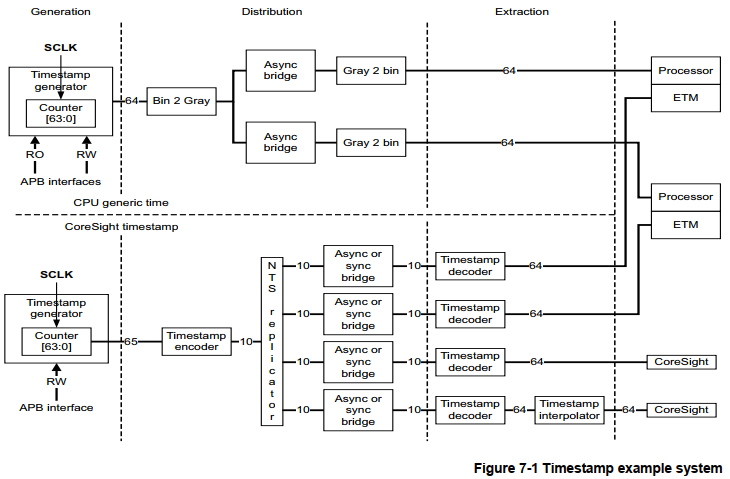
• 生成定时器事件作为中断输出。

• 事件流的生成。

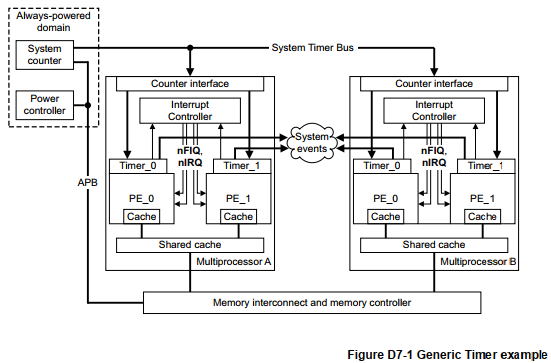
Cortex-A53 MPCore Generic Timer 符合 Arm® 架构参考手册 Armv8，适用于 Armv8-A 架构配置文件。

本章仅描述特定于 Cortex-A53 MPCore 实现的功能。

【Armv8-Generic Timer & Trace timestamp】



（图来源：DDI0480）



关于System counter:

1. 至少56-bit宽；
2. 频率：Armv8.0~Armv8.5典型1~50MHz，可以有多个可选递增模式；Armv8.6固定1GHz频率；
3. 溢出时间至少40年；
4. 精度：24小时偏差不超过10秒；
5. 从0开始启动；
6. Always-on电源域；

## 功能描述

Cortex-A53 处理器在集群的每个core中提供一组定时器寄存器。定时器是：

• 一个EL1 物理定时器。

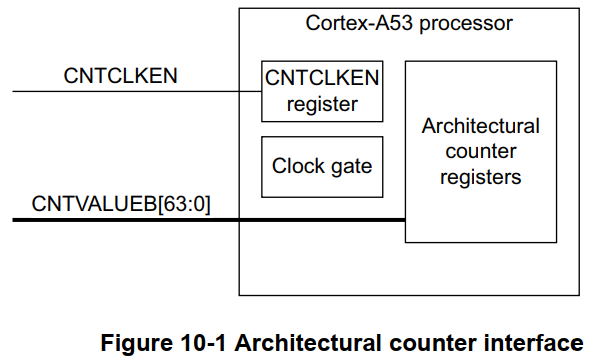
• 一个EL2 物理定时器。

• 一个EL3 物理定时器。

• 一个虚拟定时器。

Cortex-A53 处理器不包括系统计数器（system counter），在SoC中实现。系统计数器值通过同步二进制编码的64位CNTVALUEB[63:0] 分发给 Cortex-A53 处理器。

因为CNTVALUEB是从通常以低于主处理器CLKIN 的频率运行的系统计数器生成的，所以提供CNTCLKEN输入作为CNTVALUEB的时钟使能。 CNTCLKEN 在用作 CNTVALUEB[63:0] 寄存器的时钟使能之前在Cortex-A53 处理器中寄存。这允许将多周期路径（multicycle path）应用于 CNTVALUEB[63:0] 。界面如图 10-1 所示。

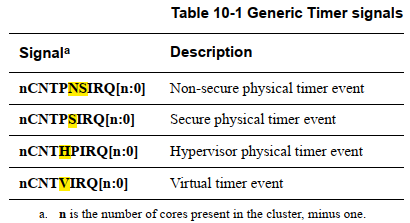


每当CNTCLKEN时钟使能的内部寄存器有效时，CNTVALUEB[63:0]上的值都需要保持稳定。 CNTCLKEN必须与CLK同步和平衡，并且必须以处理器 CLK 的整数比率切换。

有关 CNTCLKEN 的更多信息，请参见第 2-9 页的时钟。

每个定时器都向SoC提供低电平有效中断输出。

表 10-1 显示了作为外部中断输出引脚的信号。



## 定时器的寄存器

在每个core中分配了一组通用定时器寄存器。通用定时器寄存器为 32-bit宽或 64-bit宽，可在 AArch32 和 AArch64 执行状态下访问。

。。。

# Debug

## 关于

本节概述调试并描述调试组件。处理器构成调试系统的一个组件。

SoC存在以下调试Arm处理器的的方法：

**常规JTAG调试（“外部”调试）**

这是一种**侵入式**调试，使用以下方式暂停core：

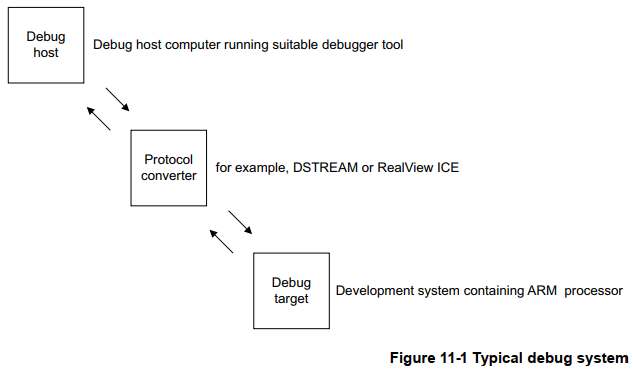
• 断点和观察点可在特定活动中暂停core。

• 用于检查和修改寄存器和memory，并提供单步执行的调试连接。

**常规监视器调试（“自托管”调试）**

这是一种**侵入式**调试，core使用驻留在memory中的调试监视器运行。

图 11-1 显示了一个典型的外部调试系统。



包含几部分：

• Debug host.

• Protocol converter.

• Debug target

• debug unit

• Self-hosted debug

### Debug host

调试主机

调试主机是一台计算机，例如个人计算机，运行诸如 DS-5 调试器之类的软件调试器。调试主机使您能够发出高级命令，例如在某个位置设置断点，或检查存储地址的内容。

### Protocol converter

协议转换器

调试主机使用以太网等接口向调试目标发送消息。但是，调试目标通常实现不同的接口协议。需要使用 DSTREAM 等设备在两种协议之间进行转换。

### Debug target

调试目标

调试目标是系统的最低层。调试目标的一个示例是带有测试芯片或带有处理器的硅部件的开发系统。

调试目标实现对协议转换器的系统支持，以使用高级外围总线 (APB) 从接口访问调试单元。

### Debug unit

调试单元

处理器调试单元协助调试在处理器上运行的软件。您可以将处理器调试单元与软件调试程序结合使用，以调试：

• 应用软件。

• 操作系统。

• 基于Arm 处理器的硬件系统。

调试单元使您能够：

• 停止程序执行。

• 检查和更改进程和协处理器状态。

• 检查和更改memory和输入/输出外设状态。

• 重新启动处理器。

### Self-hosted debug

自托管调试

对于自托管调试，调试目标运行在 Cortex-A53 处理器本身上运行的附加调试监控软件，而不需要昂贵的接口硬件来连接第二台主机计算机。

## Debug寄存器接口

处理器实现Armv8调试架构和调试事件，如 Arm® 架构参考手册 Armv8 中所述，用于 Armv8-A 架构配置文件。

调试架构定义了一组调试寄存器。调试寄存器接口提供对这些寄存器的访问：

• 处理器上运行的软件。

• 外部调试器。

本节介绍：

• 处理器接口。

• 复位对调试寄存器的影响。

• 第11-5 页的外部访问权限。

### 处理器接口

系统寄存器访问允许处理器直接访问某些调试寄存器。外部调试接口（参见第 11-36 页的外部调试接口）允许外部和自托管调试代理访问调试寄存器。对调试寄存器的访问划分如下：

**调试寄存器**

此功能是基于系统寄存器和存储映射的。您可以使用 APB slave端口访问调试寄存器映射。请参阅第 11-36 页的外部调试接口。

**性能监视器**

此功能是基于系统寄存器和存储映射的。您可以使用 APB slave端口访问性能监视器寄存器。请参阅第 11-36 页的外部调试接口。

**跟踪寄存器**

此功能是存储映射的。请参阅第 11-36 页的外部调试接口。

**交叉触发接口寄存器**

此功能是存储映射的。请参阅第 14 章交叉触发。

### 复位对调试寄存器的影响

处理器具有以下影响调试寄存器的复位信号：

**nCPUPORESET**

该信号初始化处理器逻辑，包括调试、嵌入式跟踪宏单元 (ETM) 跟踪单元、断点、观察点逻辑和性能监控逻辑。这映射到一个冷复位，包括处理器逻辑的复位和集成的调试功能。

**nCORERESET**

该信号复位一些调试和性能监控逻辑。这映射到覆盖处理器逻辑复位的热复位。

**nPRESETDBG**

该信号初始化共享调试APB、交叉触发接口 (CTI) 和交叉触发矩阵 (CTM) 逻辑。这映射到一个外部调试复位，它涵盖了外部调试接口的复位并且对处理器功能没有影响。

### 外部访问权限

对调试寄存器的外部访问权限取决于访问时的条件。表 11-1 描述了处理器对通过外部调试接口进行访问的响应。

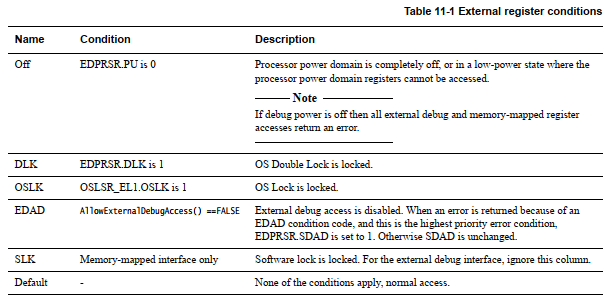
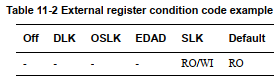


表 11-2 显示了用于访问性能监视器寄存器的外部寄存器条件代码示例。要确定寄存器的访问权限，请从左到右扫描列。在第一列停止，条件为真，条目授予寄存器的访问权限，扫描停止。



## AArch64 debug register summary

## AArch64 debug register descriptions

## AArch32 debug register summary

## AArch32 debug register descriptions

## Memory-mapped register summary

## Memory-mapped register descriptions

## Debug events

调试事件可以是：

• 软件调试事件。

• 停止调试事件。

core以下列方式之一响应调试事件：

• 忽略调试事件。

• 发生调试异常。

• 进入调试状态。

本节介绍以下方面的调试事件：

• 观察点调试事件。Watchpoint debug events.

• 调试操作系统锁。Debug OS Lock.

有关调试事件的更多信息，请参阅 Arm® 架构参考手册 Armv8，了解 Armv8-A 架构配置文件。

### 观察点调试事件

在Cortex-A53处理器中，观察点调试事件始终是同步的。Memory hint指令和cache clean操作（DC ZVA、DC IVAC 和 DCIMVAC 除外）不会生成观察点调试事件。即使排他监视器的控制检查失败，store exclusive指令也会生成观察点调试事件。

对于观察点调试事件，除了那些由cache maintenance操作产生的事件外，DFAR 中报告的值保证不低于观察点位置的地址，向下舍入为 16 byte的倍数。

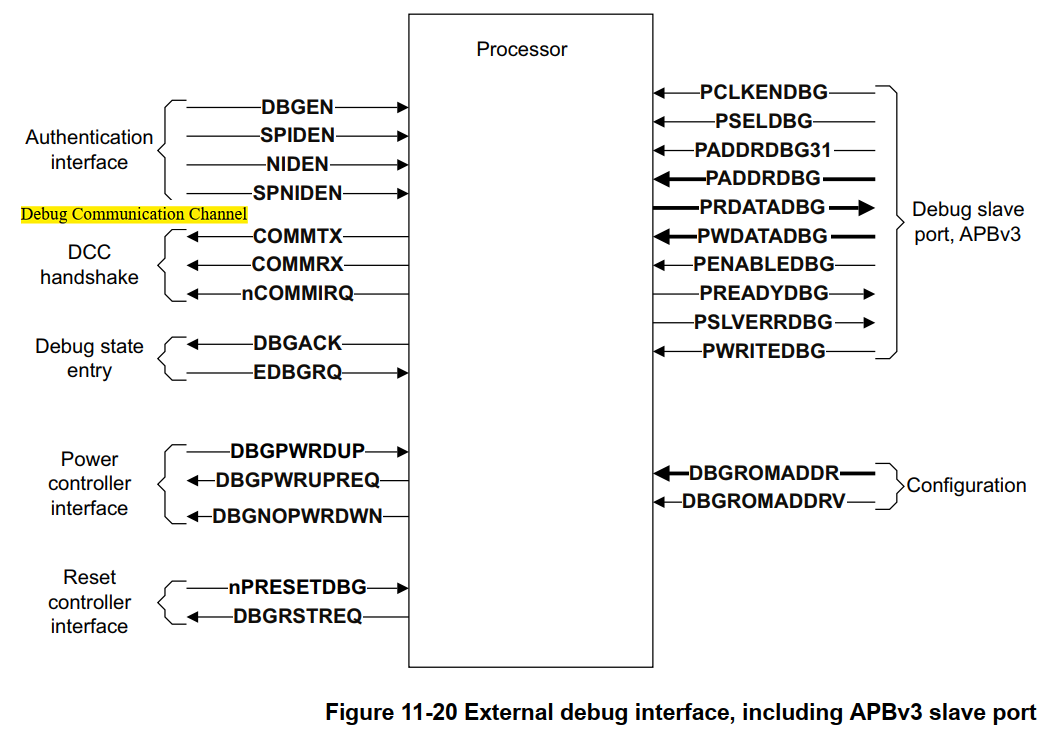
### 调试 OS 锁定

调试 OS 锁定由上电复位 nCPUPORESET 设置，请参见第 2-14 页的复位。对于调试事件和调试寄存器访问的正常行为，必须清除调试 OS 锁定。有关更多信息，请参阅 Arm® 架构参考手册 Armv8，了解 Armv8-A 架构配置文件。

## 外部调试接口

系统可以通过 APB 接口访问存储映射的调试寄存器。 APB 接口与 AMBA 4 APB 接口兼容。

图 11-20 显示了在 Cortex-A53 处理器中实现的调试接口。有关这些信号的更多信息，请参阅 Arm® CoreSight™ 架构规范。



This section describes external debug interface in:

• Debug memory map.

• DBGPWRDUP debug signal on page 11-38.

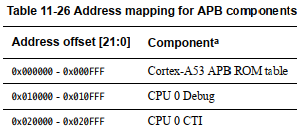
• DBGL1RSTDISABLE debug signal on page 11-38.

• Changing the authentication signals on page 11-39.

### 调试存储映射

基本存储映射最多支持集群中的四个core。表 11-26 显示了配置为 v8 调试存储映射时 Cortex-A53 处理器调试 APB 组件的地址映射。

表中的每个组件需要 4KB，并使用每个64KB 区域的底部 4KB。每个区域的剩余 60KB 被保留。



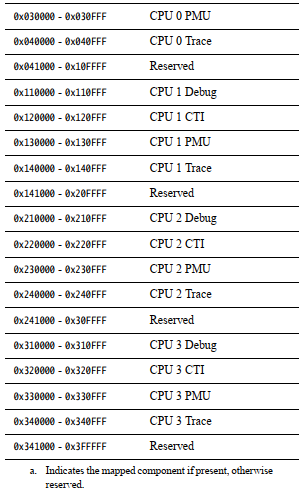
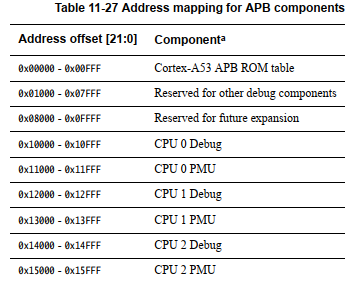
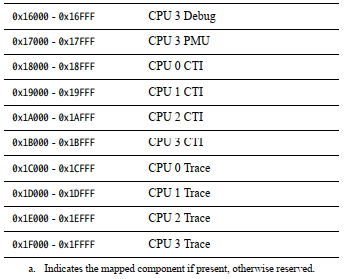


表 11-27 显示了配置为 v7 调试内存映射时 Cortex-A53 处理器调试 APB 组件的地址映射。





### DBGPWRDUP 调试信号

在断开处理器域的电源之前，您必须将DBGPWRDUP信号设置为低电平。在处理器域恢复供电后，DBGPWRDUP 信号必须置为高电平。 EDP​​RSR.PU 位反映了这个 DBGPWRDUP 信号的值。

注意

如果特定实现不支持单独的处理器和SCU电源域，则DBGPWRDUP必须绑定为高电平。

### DBGL1RSTDISABLE调试信号

当设置为HIGH 时，DBGL1RSTDISABLE 输入信号在处理器使用nCORERESET 或nCPUPORESET复位后，禁用L1 data cache的自动硬件无效化。

DBGL1RSTDISABLE 必须仅用于帮助调试外部看门狗触发的复位，方法是允许复位之前的 L1 data cache的内容在复位之后可观察。如果复位有效，则在执行 L1 data cache逐出或 L1 data cache fetch时，无法保证这些缓存条目的准确性。

您不得使用DBGL1RSTDISABLE 信号，使正常处理器上电序列中禁用 L1 data cache的自动硬件无效化。这是因为不能保证 L1 data cache失效序列与SCU中的副本 L1 tag的同步。

DBGL1RSTDISABLE 信号适用于集群中的所有core。当 nCORERESET 或 nCPUPORESET 被置位时，每个core都会对信号进行采样。

如果不需要 DBGL1RSTDISABLE 输入信号提供的功能，则输入必须连接到低电平。

### 改变认证信号

NIDEN、DBGEN、SPIDEN 和 SPNIDEN 输入信号要么绑定到某个固定值，要么由某个外部设备控制。

如果处理器上运行的软件可以控制驱动认证信号的外部设备，则它必须使用安全序列进行更改：

1. 执行特定于实现的指令序列以更改信号值。例如，这可能是一条 STR 指令，它将某些值写入系统外设中的控制寄存器。

2. 如果步骤1涉及任何store操作，则发出DSB指令。

3. 发出ISB指令或异常条目或异常返回。

4. 轮询DBGAUTHSTATUS\_EL1 以检查处理器是否已经检测到这些信号的变化值。这是必需的，因为系统可能直到DSB指令完成后的几个周期才向处理器发出信号更改。

在此过程完成之前，软件无法执行依赖于认证信号的新值的调试或分析操作。当调试器在调试状态下通过指令传输寄存器 EDITR 控制处理器时，同样的规则适用。 DBGEN、NIDEN、SPIDEN 和 SPNIDEN 值的相关组合可以通过轮询 DBGAUTHSTATUS\_EL1 来确定。

## ROM table

Cortex-A53 处理器包含一个符合 Arm® CoreSight™ 架构规范的 ROM 表。此表包含一系列组件，例如处理器调试单元、处理器交叉触发接口 (CTI)、处理器性能监控单元 (PMU) 和处理器嵌入式跟踪宏单元 (ETM) 跟踪单元。调试器可以使用 ROM 表来确定在 Cortex-A53 处理器内部实现了哪些组件。

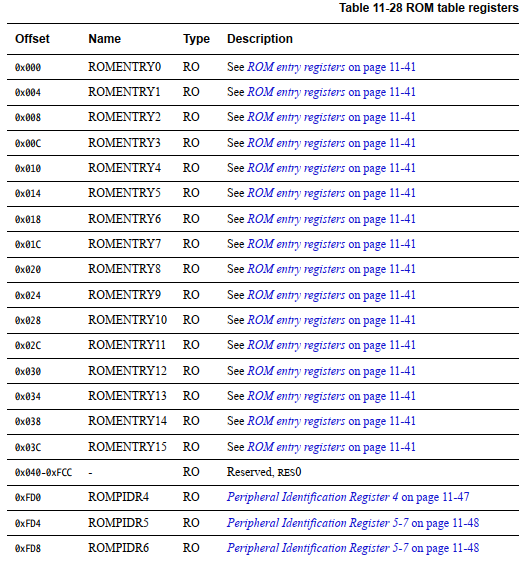
如果您的 Cortex-A53 处理器配置中不包含某个组件，则相应的调试 APB ROM 表条目仍然存在，但该组件被标记为不存在。

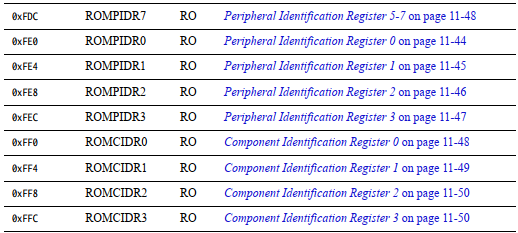
### ROM 表寄存器接口

ROM 表条目的接口是 APB slave端口。请参阅第 11-36 页的外部调试接口。

### ROM 表寄存器汇总

表 11-28 显示了与 ROM 表的物理基地址的偏移量。





# PMU

Performance Monitor Unit

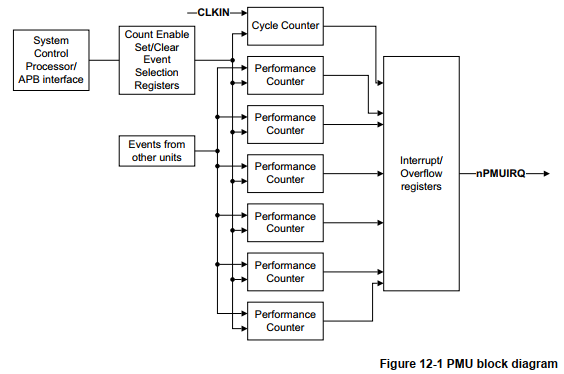
## 关于PMU

Cortex-A53处理器包括实现Arm PMUv3 架构的性能监视器。这些使您能够在运行时收集有关处理器及其内存系统操作的各种统计信息。这些提供了有关处理器行为的有用信息，您可以在调试或分析代码时使用这些信息。

PMU提供6个计数器。每个计数器都可以对处理器中可用的任何事件进行计数。由于pipeline效应，记录的绝对计数可能会有所不同。除了在很短的时间内启用计数器的情况外，这具有可以忽略不计的影响。

## PMU功能描述

PMU的主要模块如下图：



### 事件接口

来自设计中所有其他单元的事件被提供给 PMU。

### 系统寄存器和APB 接口

您可以使用系统寄存器或外部APB 接口对PMU 寄存器进行编程。

### 计数器

PMU 具有32位计数器，当它们根据事件启用时递增，还有一个64位周期计数器。

### PMU 寄存器接口

Cortex-A53 处理器支持从内部系统寄存器接口或外部调试接口访问性能监视器寄存器。请参阅第 11-36 页的外部调试界面

### 外部寄存器访问权限

PMU 寄存器的外部访问权限取决于访问时的条件。表 12-1 描述了处理器对通过外部调试接口进行访问的响应。

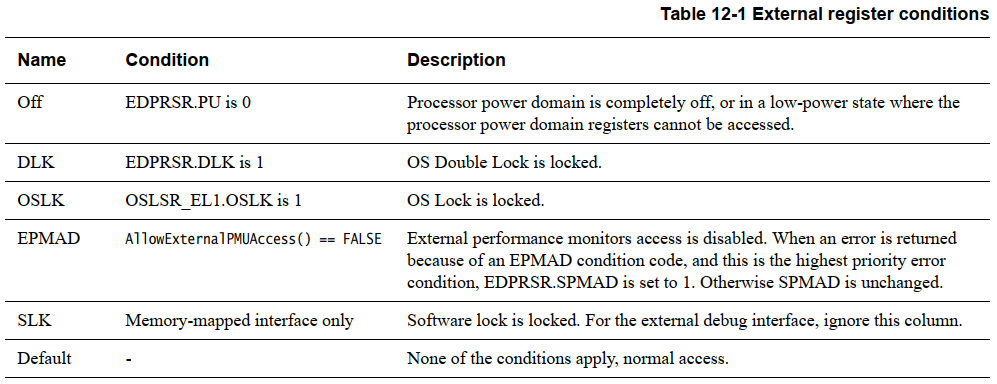
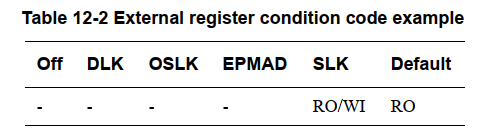


表 12-2 显示了用于访问性能监视器寄存器的外部寄存器条件代码示例。要确定寄存器的访问权限，请从左到右扫描列。在条件为真的第一列停止，该条目授予寄存器访问权限，扫描停止。



## AArch64 PMU register summary

## AArch64 PMU register descriptions

## AArch32 PMU register summary

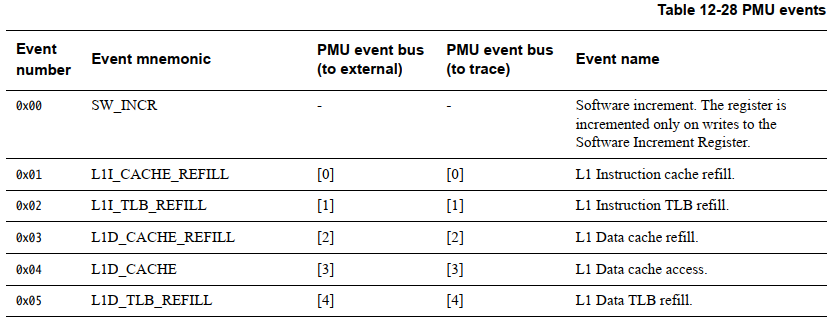
## AArch32 PMU register descriptions

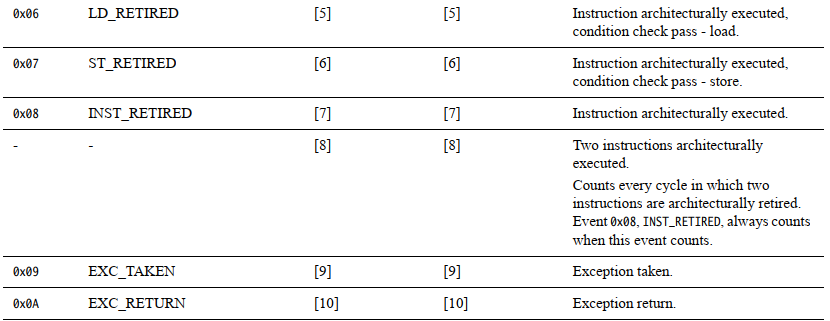
## Memory-mapped register summary

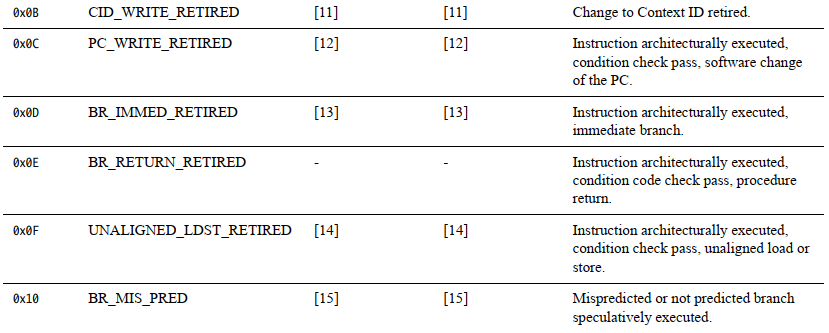
## Memory-mapped register descriptions

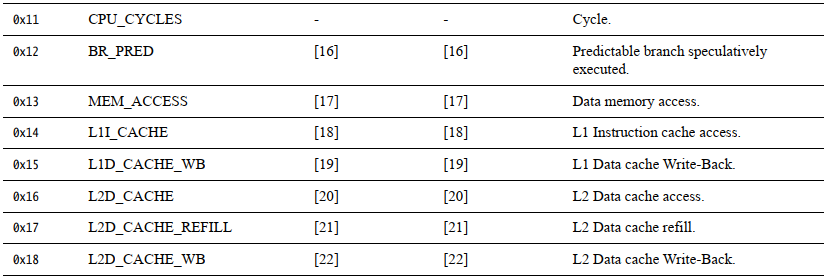
## Events

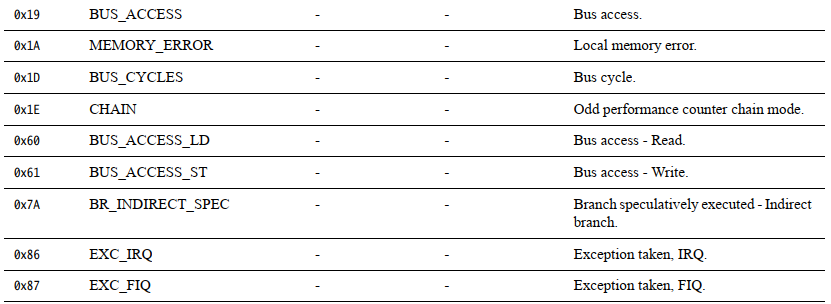
表 12-28 显示了生成的事件以及PMU用于引用事件的编号。该表还显示了**事件总线**上每个事件的bit位置。未列出的事件参考编号是保留的。

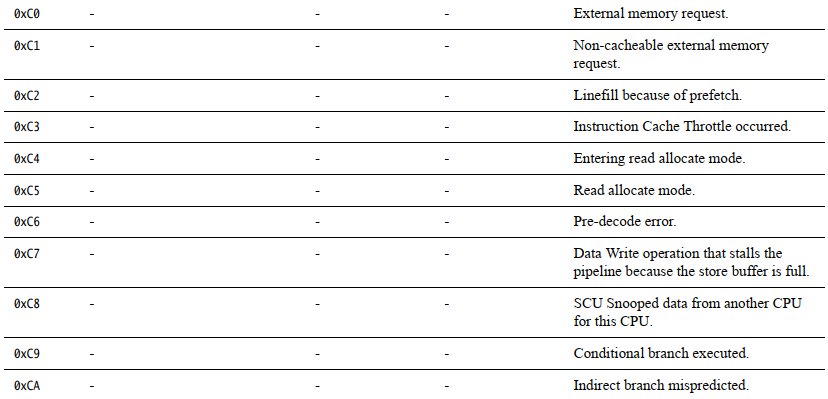


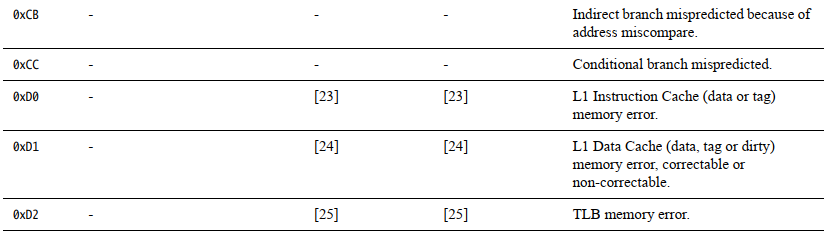


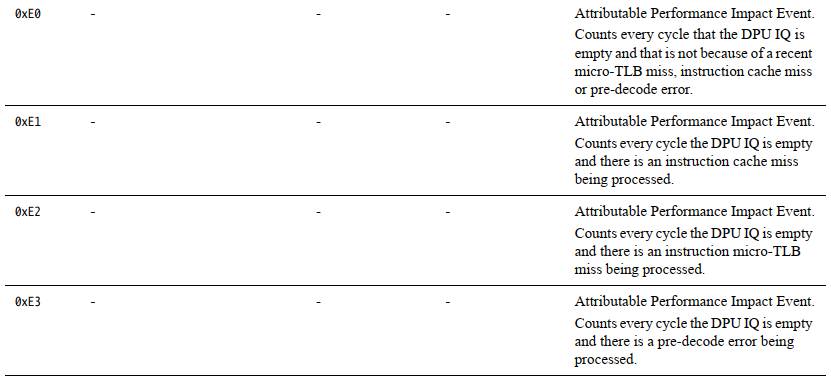


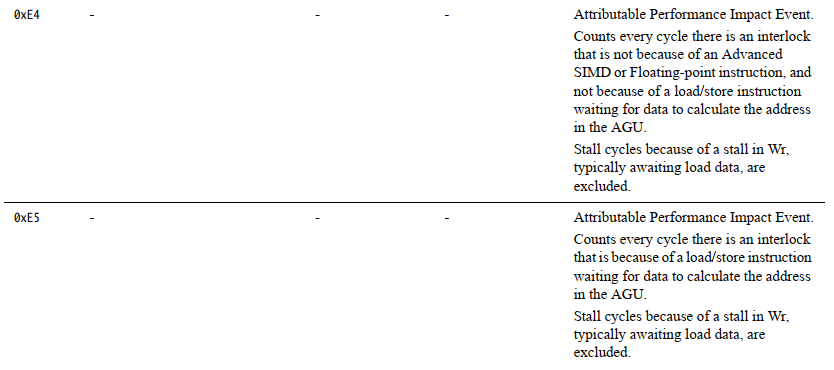


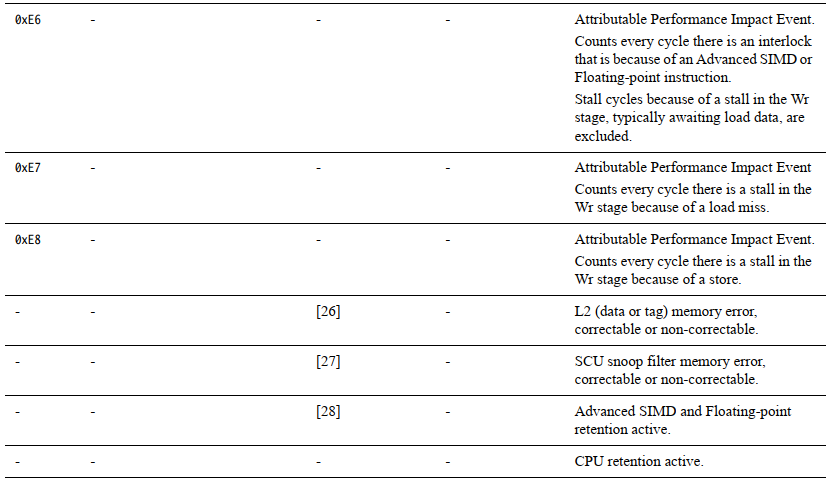












## Interrupts

当 PMU 产生中断时，Cortex-A53 处理器会置位 nPMUIRQ 信号。您可以将此信号路由到外部中断控制器以进行优先级排序和屏蔽。这是向处理器发出此中断信号的唯一机制。

该中断也作为CTI的触发输入驱动。有关详细信息，请参阅第 14 章交叉触发。

## 输出PMU事件

本节介绍如何在以下位置输出 PMU 事件：

• 外部硬件。

• 调试跟踪硬件。

### 外部硬件

除了处理器中的计数器，第 12-35 页的表 12-28 描述的一些事件在 PMUEVENT 总线上输出，可以连接到外部硬件。

### 调试跟踪硬件

第 12-35 页的表 12-28 描述的一些事件被输出到 ETM 跟踪单元或其他外部跟踪硬件，以启用事件监控。有关详细信息，请参阅第 13 章嵌入式跟踪宏单元和第 14 章交叉触发。

# ETM

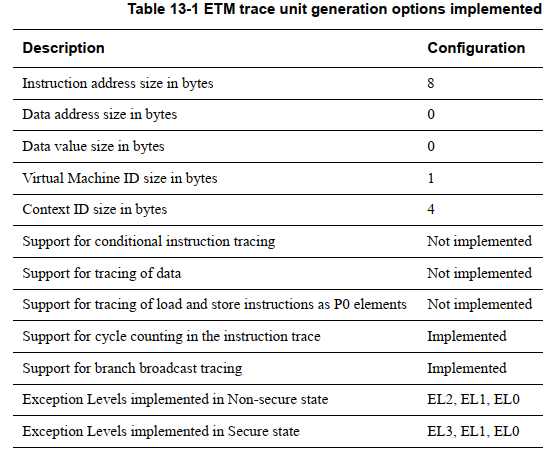
Embedded trace macrocell

## 关于ETM

ETM跟踪单元是基于嵌入式跟踪宏单元（ETM，Embedded Trace Macrocell）架构ETMv4进行实时指令流跟踪的模块。 ETM 是CoreSight组件，是Arm实时调试解决方案DS-5 Development Studio的组成部分。有关详细信息，请参阅第 ix 页的附加阅读中的 CoreSight 文档。

## 生成选项和源

表 13-1 显示了在 Cortex-A53 ETM 跟踪单元中实现的跟踪生成选项。



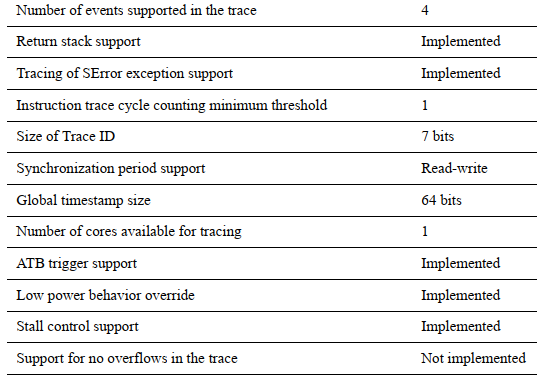
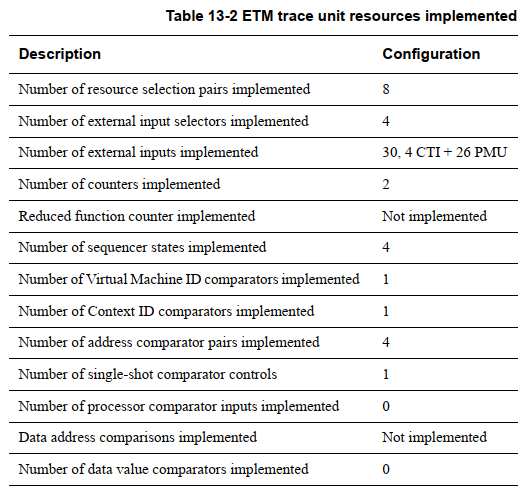


表 13-2 显示了 Cortex-A53 ETM 跟踪单元中实现的资源。



## 功能描述

本节介绍 ETM 跟踪单元。它包含以下部分：

• 处理器接口。

• 跟踪生成。

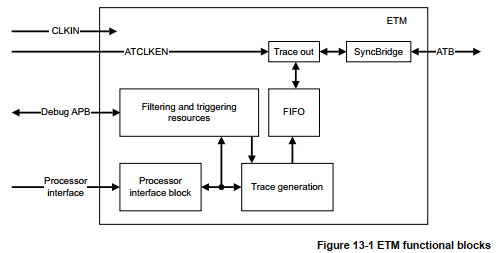
• 过滤和触发源。

• 先进先出。

• 跟踪输出

• 同步桥

图 13-1 显示了 ETM 跟踪单元的主要功能块。



### 处理器接口

该模块监控处理器的行为并生成 P0 元素，这些元素本质上是执行的指令和按程序顺序跟踪的异常。

### 跟踪生成

跟踪生成块根据 P0 元素生成各种跟踪包。

### 过滤和触发资源

您可以通过过滤过程限制ETM 生成的跟踪数据量。例如，仅在某个地址范围内生成跟踪。还提供更复杂的逻辑分析仪样式过滤选项。

ETM 跟踪单元还可以生成一个触发器，该触发器是跟踪捕获设备停止捕获跟踪的信号。

### FIFO

ETM 跟踪单元生成的跟踪是高度压缩的形式。 FIFO 使跟踪突发被展平。当 FIFO 变满时，FIFO 发出溢出信号。在 FIFO 被清空之前，跟踪生成逻辑不会生成任何新的跟踪。在调试器中查看时，这会导致跟踪出现间隙。

### Trace out

Trace from FIFO 在同步 AMBA ATB 接口上输出。

### Syncbridge

来自trace out 模块的ATB 接口通过一个ATB 同步桥。

## 复位

ETM 跟踪单元的复位与处理器的冷复位相同。当对处理器应用热复位时，ETM 跟踪单元不会复位，因此可以通过热处理器复位进行跟踪。

如果 ETM 跟踪单元被复位，跟踪将停止，直到 ETM 跟踪单元被重新编程并重新启用。但是，如果使用热复位对处理器进行复位，则可能无法跟踪复位前处理器提供的最后几条指令。

## Modes of operation and execution

## ETM trace unit register interfaces

## ETM register summary

## ETM register descriptions

## 与调试和性能监控单元的交互

本节描述：

• 与性能监控单元的交互。

• 调试双重锁定对跟踪寄存器访问的影响。

### 与性能监控单元的交互

Cortex-A53 处理器包括一个性能监控单元 (PMU)，它可以在一段时间内对事件（例如缓存未命中和执行的指令）进行计数。有关详细信息，请参阅第 12 章性能监视器单元。本节介绍 PMU 和 ETM 跟踪单元如何协同工作。

**ETM 跟踪单元对 PMU 事件的使用**

所有 PMU 体系结构事件都可通过扩展输入工具提供给 ETM 跟踪单元。有关 PMU 事件的更多信息，请参阅 Arm® 架构参考手册 Armv8，了解 Armv8-A 架构配置文件。

ETM 跟踪单元使用四个扩展的外部输入选择器来访问 PMU 事件。每个选择器可以独立地选择一个 PMU 事件，然后在相关事件发生的周期中激活这些事件。然后可以通过 ETM 跟踪单元中的任何事件寄存器访问这些选定的事件。第 12-35 页的表 12-28 描述了 PMU 事件。

### 调试双重锁定对跟踪寄存器访问的影响

通过外部调试接口进行的所有跟踪寄存器访问都表现为当设置了调试双重锁定时处理器电源域已断电。有关调试双锁的更多信息，请参阅 Arm® 架构参考手册 Armv8，了解 Armv8-A 架构配置文件。

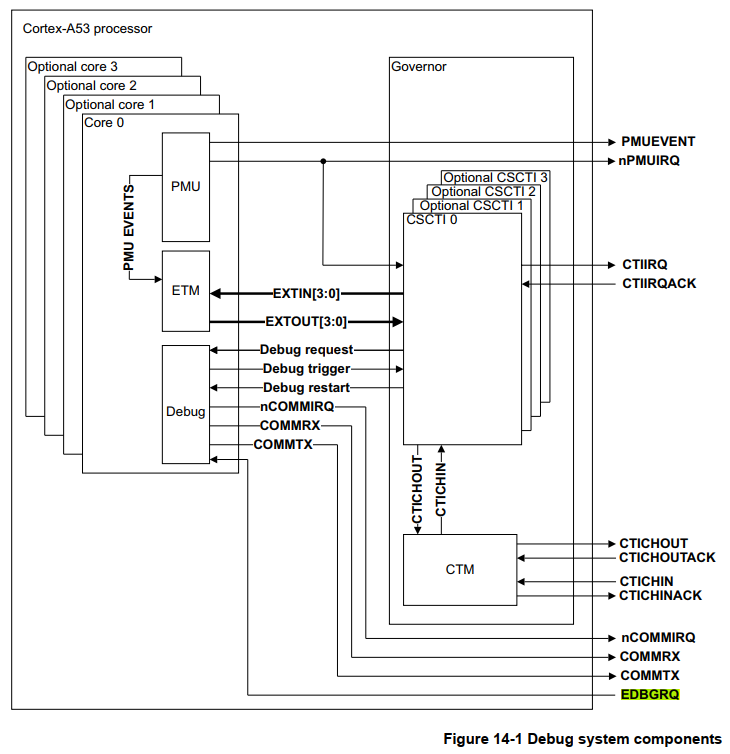
# Cross Trigger

## 关于Cross Trigger

Cortex-A53 处理器具有单个外部交叉触发通道接口。这个外部接口通过一个Cross Trigger Matrix (CTM) 连接到每个核心对应的CoreSight Cross Trigger Interface (CTI) 接口。许多Embedded Cross Trigger (ECT) 触发输入和触发输出连接在Cortex-A53和CoreSight CTI模块中的调试组件之间。

CTI使调试逻辑、ETM跟踪单元和PMU能够相互交互并与其他CoreSight组件交互。这称为交叉触发。例如，将CTI配置为在ETM跟踪单元触发事件发生时生成中断。

图 14-1 显示了调试系统组件以及可用的触发输入和触发输出。



## Trigger输入和输出

描述CTI的输入

|  |  |  |
| --- | --- | --- |
| CTI input | 名称 | 描述 |
| 0 | DBGTRIGGER, pulsed | 脉冲进入调试状态 |
| 1 | PMUIRQa | PMU产生的中断 |
| 2 | - | - |
| 3 | - | - |
| 4 | EXTOUT[0] | ETM外部输出 |
| 5 | EXTOUT[1] | ETM外部输出 |
| 6 | EXTOUT[2] | ETM外部输出 |
| 7 | EXTOUT[3] | ETM外部输出 |

a. This signal is the same as nPMUIRQ with inverted polarity.

描述CTI的输出

|  |  |  |
| --- | --- | --- |
| CTI output | 名称 | 描述 |
| 0 | EDBGRQ | 使处理器进入调试状态 |
| 1 | DBGRESTART | 使处理器退出调试状态 |
| 2 | CTIIRQ | CTI中断 |
| 3 | - | - |
| 4 | EXTIN[0] | ETM外部输入 |
| 5 | EXTIN [1] | ETM外部输入 |
| 6 | EXTIN [2] | ETM外部输入 |
| 7 | EXTIN [3] | ETM外部输入 |

## CTM

来自所有core的CoreSight CTI 通道信号使用Cross Trigger Matrix (CTM) 模块进行组合，以便在 Cortex-A53 处理器中呈现单个交叉触发通道接口。该模块最多可以将四个内部通道接口（每个Core对应一个）与一个外部通道接口组合在一起。

在 Cortex-A53 CTM 中，外部通道输出由所有内部通道输出的OR输出驱动。除了外部通道输入之外，每个内部通道输入都由所有其他 CTI 的内部通道输出的OR输出驱动。

## Cross trigger register summary

## Cross trigger register descriptions

# A 信号描述

## 前言

本附录中的表格列出了 Cortex-A53 处理器信号，以及它们的方向：输入或输出，以及高级描述。

一些总线包括一个可配置的宽度字段 <Signal>[CN:0]，其中CN = 0、1、2 或 3，最多可编码四个core。例如：

• nIRQ[0] 表示core 0 中断请求。

• nIRQ[2] 代表core 2 中断请求。

某些信号以 <signal>x 的形式指定，其中 x = 0、1、2 或 3 以reference core 0、core 1、core 2、core 3。如果不存在core，则移除相应的引脚。例如：

• PMUEVENT0[29:0] 代表core 0 PMU事件总线。

• PMUEVENT3[29:0] 代表core 3 PMU 事件总线。

信号的数量根据配置而变化。例如，当处理器配置为具有AMBA 4 ACE接口时，不存在 AMBA 5 CHI接口信号。

## 时钟

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 方向 | 描述 | 集成处理 |
| CLKIN | Input | Global clock | SUB\_CRG  核的主时钟，当前目标频率1.8~2.0G  内部各时钟分别通过：  PCLKENDBG/ ACLKENM/ ACLKENS/ SCLKEN/ ATCLKEN/ CNTCLKEN进行分频，时钟使能处理 |
| PCLKENDBG | Input | APB clock enable. | SUB\_CRG |
| ACLKENM | Input | ACE Master bus clock enable. See Clocks on page 2-9 for more information. | SUB\_CRG |
| ~~SCLKEN~~ | ~~input~~ | ~~CHI接口时钟使能~~ | ~~不涉及~~ |
| ATCLKEN | Input | ATB clock enable | SUB\_CRG |
| ACLKENS | Input | AXI slave bus clock enable. | SUB\_CRG |
| CNTCLKEN | Input | Counter clock enable.  该时钟必须在CNTVALUEB之前一个周期有效 | SUB\_CRG  ALLON频率？ |

## 复位

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 方向 | 描述 | 集成处理 |
| nCPUPORESET[CN:0] | Input | Processor powerup reset:  0 对处理器所有逻辑复位a.  1 Do not apply reset to all processor logica. | SUB\_CRG  上电复位信号，每个core可以分开控制，复位范围整个core  通过top crg进行控制。 |
| nCORERESET[CN:0] | Input | Individual core resets excluding Debug and ETM trace unit:  0 对处理器逻辑复位b.  1 Do not apply reset to processor logicb. | SUB\_CRG  在初始化的时候，主核先解复位，其他核该复位信号保持  各复位要求>10个cycle，保守>18个cycle |
| nL2RESET | Input | L2存储系统复位：  0 对共享L2存储系统控制器复位  1 Do not apply reset to shared L2 memory system controller. | SUB\_CRG  针对L2 memory system的软复位 |
| L2RSTDISABLE | Input | 在复位时关闭L2 Cache自动invalidate:  0 硬件复位L2 Cache  1硬件不复位L2 Cache | 该信号通过global sysctrl进行控制，默认可以选择为0 |
| WARMRSTREQ[CN:0] | Output | Processor热复位请求  0 Do not apply warm reset.  1 Apply warm reset.  （建议：接到reset control） | 该信号接到top crg或lpc中，用于产生core的软复位 |
| nPRESETDBG | Input | APB reset, active-LOW:  0 Apply reset to APB interface.  1 Do not apply reset to APB interface. | SUB\_CRG  reset control |
| nMBISTRESET | Input | MBIST reset | SUB\_CRG  reset control |
| DBGRSTREQ[CN:0] | Output | Warm reset request.  （建议：接到reset control） | SUB\_CRG  reset control |
| DBGL1RSTDISABLE | Input | 关闭L1 data cache在复位时自动无效化：  0 开启自动无效化  1 关闭自动无效化  该引脚只在处理器复位时被采样。  （只在调试时可关闭，正常使用过程，该信号必须接低电平） | 该信号通过global sysctrl进行控制，默认为0 |

a. Processor logic包括高级SIMD和FP、Debug、ETM、breakpoint和watchpoint逻辑

b. Processor logic包括高级SIMD和FP、但不包括Debug、ETM、breakpoint和watchpoint逻辑

【L2】

nL2RESET：控制对L2系统可复位寄存器进行复位；

L2RSTDISABLE：控制解复位后L2 Cache自动invalidation；

L2FLUSHREQ：要进入cluster低功耗前，将L2 cache的数据刷回DDR；刷完成后对应的L2FLUSHDONE拉起。

## 配置

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 方向 | 描述 | 集成处理 |
| AA64nAA32[CN:0] | Input | Register width state:  0 AArch32.  1 AArch64.  该引脚只在处理器复位时被采样。 | 系统控制器分开可配控制，默认值1’b1  64启动可转32，但32不可转64 |
| CFGEND[CN:0] | Input | 复位时大小端配置，用来初始化CP15 SCTLR\_EL3和SCTR\_S寄存器的EE bit:  0 EE bit is LOW.（小端）  1 EE bit is HIGH.  该引脚只在处理器复位时被采样。 | 固结4'b0000 |
| CFGTE[CN:0] | Input | 使能T32异常，用来初始化CP15 SCTLR寄存器的TE bit:  0 TE bit is LOW. （在A32状态下产生异常）  1 TE bit is HIGH.（在T32状态下产生异常）  该引脚只在处理器复位时被采样。  （异常处理使用的指令集是Arm指令集还是Thumb2指令集？） | 系统控制器分开可配控制，默认值1’b0  arm32指令处理execption |
| CLUSTERIDAFF1[7:0] | Input | CP15 MPIDR寄存器中，Cluster亲和ID Level 1的值MPIDR bits[15:8]。  该引脚只在处理器复位时被采样。  （多cluster系统接不同的值。如果只有一个cluster，则固结低电平。） | 0x00 |
| CLUSTERIDAFF2[7:0] | Input | CP15 MPIDR寄存器中，Cluster亲和ID Level 1的值MPIDR bits[23:16]。  该引脚只在处理器复位时被采样。  （多cluster系统接不同的值。如果只有一个cluster，则固结低电平。） | 0x00 |
| CP15SDISABLE[CN:0] | Input | 关闭对安全CP15寄存器的写访问。  （接高电平，禁止访问安全CP15寄存器。接低电平可访问所有CP15寄存器） | 系统控制器控制，默认disable，必须安全访问才能修改该配置，默认值0 |
| CRYPTODISABLE[CN:0] | Input | 关闭加解密扩展。  该引脚只在处理器复位时被采样。 | 固结为0，使能CRYPTOGRAPHY |
| RVBARADDRx[39:2] | Input | 在64-bit执行态下的复位向量基地址。（启动地址，RVBAR\_EL3）  该引脚只在处理器复位时被采样。 | gsys启动地址，  默认值Bootrom的启动地址，待系统memorymap确定； |
| VINITHI[CN:0] | Input | 复位时异常向量，初始化CP15 SCTLR寄存器V bit。  0 异常向量起始地址为0x00000000.  1 异常向量起始地址为0xFFFF0000.  该引脚只在处理器复位时被采样。 | 固结为0，DDR的起始地址 |

## 通用中断控制器信号

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 方向 | 描述 | 集成处理 |
| nFIQ[CN:0] | Input | FIQ请求，低有效，电平敏感，异步中断请求：  0 Activate FIQ interrupt.  1 Do not activate FIQ interrupt.  处理器将nFIQ输入作为电平敏感处理，nFIQ有效要持续到处理器对中断进行确认。 | 系统控制器，默认值4’b1111  中断使用GIC CPU interface |
| nIRQ[CN:0] | Input | IRQ请求，低有效，电平敏感，异步中断请求：  0 Activate interrupt.  1 Do not activate interrupt.  处理器将nIRQ输入作为电平敏感处理，nIRQ有效要持续到处理器对中断进行确认。 | 系统控制器，默认值4’b1111  中断使用GIC CPU interface |
| nSEI[CN:0] | Input | 系统错误中断请求（System Error Interrupt），低有效，边沿敏感：  0 Activate SEI request.  1 Do not activate SEI request.  处理器将nSEI作为边沿敏感处理，nSEI信号必须以脉冲的方式输入给处理器。  nSEI置位会产生：  • Asynchronous Data Abort（如果是AArch32），DFSR.FS域被置位，表示该异常。  • SError interrupt（如果是AArch64），ESR\_ELx.ISS域被置位，表示该中断。 | 系统控制器，默认值4’b1111  中断使用GIC CPU interface |
| nVFIQ[CN:0] | Input | Virtual FIQ请求，低有效，电平敏感，异步中断请求：  0 Activate FIQ interrupt.  1 Do not activate FIQ interrupt.  处理器将nVFIQ输入作为电平敏感处理，nVFIQ有效要持续到处理器对中断进行确认。  如果启用了GIC（GICCDISABLE信号接低电平），则nVFIQ 输入必须接死为高电平。  如果未启用了GIC（GICCDISABLE信号接高电平），则nVFIQ 输入由SOC系统的GIC驱动。 | 固结4’b1111  中断使用GIC CPU interface |
| nVIRQ[CN:0] | Input | Virtual IRQ请求，低有效，电平敏感，异步中断请求：  0 Activate interrupt.  1 Do not activate interrupt.  处理器将nVIRQ输入作为电平敏感处理，nVIRQ有效要持续到处理器对中断进行确认。  如果启用了GIC（GICCDISABLE信号接低电平），则nVIRQ 输入必须接死为高电平。  如果未启用了GIC（GICCDISABLE信号接高电平），则nVIRQ 输入由SOC系统的GIC驱动。 | 固结4’b1111  中断使用GIC CPU interface |
| nVSEI[CN:0] | Input | 虚拟系统错误中断请求（Virtual System Error Interrupt），低有效，边沿敏感：  0 Activate virtual SEI request.  1 Do not activate virtual SEI request.  处理器将nVSEI作为边沿敏感处理，nVSEI信号必须以脉冲的方式输入给处理器。  nVSEI置位会产生：  • Asynchronous Data Abort（如果是AArch32），DFSR.FS域被置位，表示该异常。  • SError interrupt（如果是AArch64），ESR\_EL1.ISS域被置位，表示该中断。 | 固结4’b1111  中断使用GIC CPU interface |
| nREI[CN:0] | Input | RAM错误中断请求，低有效，边沿敏感：  0 Activate REI request. 报告系统中的RAM错误  1 Do not activate REI request.  处理器将nREI作为边沿敏感处理，nREI信号必须以脉冲的方式输入给处理器。  nREI置位会产生：  • Asynchronous Data Abort（如果是AArch32），DFSR.FS域被置位，表示该异常。  • SError interrupt（如果是AArch64），ESR\_ELx.ISS域被置位，表示该中断。 | 固结4’b1111  中断使用GIC CPU interface |
| nVCPUMNTIRQ[CN:0] | Output | 虚拟CPU接口维护中断PPI输出。 | 对接GIC-PPI-25 |
| PERIPHBASE[39:18] | Input | 为GIC寄存器指定基地址，该值在复位时被采样到CP15 CBAR寄存器。 | 接memory map GIC地址，方便软件读取访问GIC存储映射寄存器 |
| GICCDISABLE | Input | 关闭GIC CPU interface逻辑，并路由“External”信号直接给处理器：  0 使能GIC CPU interface  1 关闭GIC CPU interface，并路由nIRQ, nFIQ,nVIRQ, nVFIQ 直接给处理器。  当使用GIC-400时，将该信号置为高，其不支持GICv3/v4。  （该引脚只在处理器复位退出时被采样。） | 固结0  使用GIC CPU interface |
| GICC Stream接口 | | | 对接A53 GIC interface |
| ICDTVALID | Input | AXI4-Stream协议信号. Distributor到GIC CPU Interface 消息的信号 TVALID，指示消息有效 | - |
| ICDTREADY | Output | AXI4-Stream协议信号. Distributor到GIC CPU Interface 消息的信号TREADY，指示slave当拍可以接收一次传输 | - |
| ICDTDATA[15:0] | Input | AXI4-Stream协议信号. Distributor到GIC CPU Interface 消息的信号TDATA，指示消息内容 | - |
| ICDTLAST | Input | AXI4-Stream协议信号. Distributor到GIC CPU Interface 消息的信号TLAST，指示包边界 | - |
| ICDTDEST[1:0] | Input | AXI4-Stream协议信号. Distributor到GIC CPU Interface 消息的信号TDEST，提供数据流路由信息  取决于cluster的配置，不一定使用所有bit：  00 Core1  01 Core2  10 Core3  11 Core4  MP1 两bit接LOW  MP2 bit[1]接LOW  MP3,MP4 两bit都用 | - |
| ICCTVALID | Output | AXI4-Stream协议信号. GIC CPU Interface到Distributor消息的信号TVALID，指示消息有效 | - |
| ICCTREADY | Input | AXI4-Stream协议信号. GIC CPU Interface到Distributor消息的信号TREADY，指示slave当拍可以接收一次传输 | - |
| ICCTDATA[15:0] | Output | AXI4-Stream协议信号. GIC CPU Interface到Distributor消息的信号TDATA，指示消息内容 | - |
| ICCTLAST | Output | AXI4-Stream协议信号. GIC CPU Interface到Distributor消息的信号TLAST，指示包边界 | - |
| ICCTID[1:0] | Output | AXI4-Stream协议信号. GIC CPU Interface到Distributor消息的信号TID，指示不同数据流的标识  00 Core1  01 Core2  10 Core3  11 Core4 | - |

## 通用定时器

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 方向 | 描述 | 集成处理 |
| nCNTHPIRQ[CN:0] | Output | Hypervisor physical timer event. | GIC-PPI-26 |
| nCNTPNSIRQ[CN:0] | Output | Non-secure physical timer event. | GIC-PPI-30 |
| nCNTPSIRQ[CN:0] | Output | Secure physical timer event. | GIC-PPI-29 |
| nCNTVIRQ[CN:0] | Output | Virtual physical timer event. | GIC-PPI-27 |
| CNTCLKEN | Input | 见时钟部分 | - |
| CNTVALUEB[63:0] | Input | 全局系统计数器二进制格式值。  (可以将timestamp生成器的计数器值接过来) | gsys，64-bit free running counter  allon 计数器，格雷码传到本模块后转bin码后接到A53 |

## 电源管理

Non-Retention电源管理

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 方向 | 描述 | 集成处理 |
| CLREXMONREQ | Input | 清除external global exclusive monitor的请求，产生一个WFE唤醒事件到所有core。见2.4.2.2  （接到全局排他监视器，如果其不存在，则该信号接LOW） | 作为WFE唤醒请求，固结为1'b0 |
| CLREXMONACK | Output | 清除外部全局排他monitor的确认。见2.4.2.2  （接到全局排他监视器） | 悬空 |
| EVENTI | Input | 处理器从WFE状态唤醒的事件输入，见2.4.3  （接到power controller） | power controller，默认值0 |
| EVENTO | Output | 事件输出，当执行SEV指令时该信号有效，见2.4.3  （接到power controller）  事件输出，当执行SEV指令时该信号有效，一个core通知其他core唤醒，同时输出该脉冲信号 | power controller |
| STANDBYWFI[CN:0] | Output | 指示core是否处于WFI低功耗状态：  0 Core not in WFI low-power state.  1 Core in WFI low-power state. This is the reset condition.  （接到power controller，电源控制器在观测到core的该信号位LOW时，对应core的电源不可移除） | power controller  送到系统控制器做记录 |
| STANDBYWFE[CN:0] | Output | 指示core是否处于WFE低功耗状态：  0 Core not in WFE low-power state.  1 Core in WFE low-power state.  （接到power controller） | power controller  送到系统控制器做记录 |
| STANDBYWFIL2 | Output | 指示L2存储系统是否处于WFI低功耗状态，当满足如下条件时，该信号有效：  • 所有core都处于WFI,或复位保持，或nl2RESET置位。  • ACE配置下，将ACINACTM置高电平  • CHI配置下，将SINACT 置高电平  • ACP 配置下，将AINACTS置高电平  • L2存储系统空闲  （接到power controller，电源控制器在观测到该信号为LOW时，不可对cluster移除电源） | power controller  送到系统控制器做记录 |
| L2FLUSHREQ | Input | L2硬件刷新请求  （接到power controller） | power controller，默认值0。  或接到系统控制器，默认值为0 |
| L2FLUSHDONE | Output | L2 硬件刷新完成  （接到power controller） | power controller，或接到系统控制器可读 |
| SMPEN[CN:0] | Output | 指示一个core是否纳入了一致性中。  （CPUECTLR\_EL1.SMPEN，使能cache功能前，要将该寄存器写1）单核下电流程使用 | power controller  送到系统控制器做记录 |
| DBGNOPWRDWN[CN:0] | Output | Core不下电请求：  0 Do not request that the core stays powered up.  1 Request that the core stays powered up.  （接到system power controller，系统电源控制器在观测到core该信号为HIGH时，对应core或memory不能移除电源，即使在powerdown envent期间） | power controller  送到系统控制器做记录 |
| DBGPWRUPREQ[CN:0] | Output | Core上电请求:  0 Do not request that the core is powered up.  1 Request that the core is powered up.  （接到system power controller，该请求由debug agent发出，将对应core进行上电） | power controller  送到系统控制器做记录 |
| DBGPWRDUP[CN:0] | Input | Core已上电  0 Core is powered down.  1 Core is powered up.  （接到system power controller，指示对应的PDCPUn域已上电。  如果不支持PDCORTEXA53和PDCPUn分开控制，则该信号接HIGH） | 固结4’b1111  送到系统控制器做记录 |

Retention电源管理

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 方向 | 描述 | 集成处理 |
| CPUQACTIVE[CN:0] | Output | 指示core是否活动  （接到power controller） | 悬空 |
| CPUQREQn[CN:0] | Input | 指示电源控制器准备为core进入或退出retention  （接到power controller，或不使用接HIGH） | 固结1 |
| CPUQDENY[CN:0] | Output | 指示core拒绝电源控制器的retention请求  （接到power controller） | 悬空 |
| CPUQACCEPTn[CN:0] | Output | 指示core接受电源控制器的retention请求  （接到power controller） | 悬空 |
| NEONQACTIVE[CN:0] | Output | 指示高级SIMD浮点部件是否活动  （接到power controller） | 悬空 |
| NEONQREQn[CN:0] | Input | 指示电源控制器准备为高级SIMD浮点部件进入或退出retention  （接到power controller，或不使用接HIGH） | 固结1 |
| NEONQDENY[CN:0] | Output | 指示高级SIMD浮点部件拒绝电源控制器的retention请求  （接到power controller） | 悬空 |
| NEONQACCEPTn[CN:0] | Output | 指示高级SIMD浮点部件接受电源控制器的retention请求  （接到power controller） | 悬空 |
| L2QACTIVE | Output | 指示L2 Data RAMs是否活动 | 悬空 |
| L2QREQn | Input | 指示电源控制器准备为L2 Data RAMs进入或退出retention  （接到power controller，或不使用接HIGH） | 固结1 |
| L2QDENY | Output | 指示L2 Data RAMs拒绝电源控制器的retention请求  （接到power controller） | 悬空 |
| L2QACCEPTn | Output | 指示L2 Data RAMs接受电源控制器的retention请求  （接到power controller） | 悬空 |

## L2 error

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 方向 | 描述 | 集成处理 |
| nEXTERRIRQ | Output | AXI或CHI传输时有错误响应的错误指示。  （接到中断控制器） | GIC-1-N（SPI） |
| nINTERRIRQ | Output | L2 RAM两bit ECC错误指示  （接到中断控制器） | GIC-1-N（SPI） |

## ACE和CHI接口

公共信号

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 方向 | 描述 | 集成处理 |
| BROADCASTCACHEMAINT | Input | 使能广播cache维护操作到下游cache:  0 Cache维护操作不可广播到下游cache  1 Cache维护操作可广播到下游cache  该引脚只在处理器复位时被采样。 | 固结为0，系统没有L3 cache |
| BROADCASTINNER | Input | 使能广播inner shareable传输：  0 Inner Shareable传输不广播到外部  1 Inner Shareable传输广播到外部  如果BROADCASTINNER接高电平，则BROADCASTOUTER必须接高电平  该引脚只在处理器复位时被采样。 | 固结为0，系统中没有多个cpu cluster，不需要考虑cpu cluster间硬件一致性 |
| BROADCASTOUTER | Input | 使能广播outer shareable传输：  0 Outer Shareable传输不能广播到外部  1 Outer Shareable传输可以广播到外部  该引脚只在处理器复位时被采样。 | 固结为0，无out-coherency。 |

## CHI接口

Cortex-A53配置为CHI接口时才有该部分信号。

**时钟和配置**

**TXREQ、TXRSP、TXDAT、RXSNP、RXRSP、RXDAT**

**系统地址映射**

## ACE接口

Cortex-A53配置为ACE接口时才有该部分信号。

All ACE channels must be balanced with respect to CLKIN and timed relative to ACLKENM.

**时钟和配置**

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 方向 | 描述 | 集成处理 |
| ACLKENM | Input | 见时钟部分 | - |
| ACINACTM | Input | Snoop接口不活动，且不参与一致性：  0 Snoop interface is active.  1 Snoop interface is inactive.  （如果该信号接LOW，处理器会禁止STANDBYWFIL2置位。在单cluster系统，该信号接HIGH，使其表现为AXI3兼容模式） | ACE mode，固结为0，使用snoop接口 |
| SYSBARDISABLE | Input | 关闭向系统总线广播barrier操作：  0 Barriers广播到系统总线，需要是AMBA4 ACE或AMBA5 CHI,总线  1 Barriers不能广播到系统总线，兼容AXI3和AMBA4互连.a  该引脚只在处理器复位时被采样。.  （在多cluster环境，可以连接该信号通知下游广播barrier操作。若使用AXI3兼容模式，则该信号接HIGH。使用CHI时，接LOW） | A53 TRM建议将barrier操作在Cluster内部终结掉，以获得最佳性能。  固结1 |
| RDMEMATTR[7:0] | Output | Read request memory attributes. | 悬空 |
| WRMEMATTR[7:0] | Output | Write request memory attributes. | 悬空 |

a. See Barriers on page 7-11

**AW通道**

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 方向 | 描述 | 集成处理 |
| AWADDRM[43:0] | Output | Write address. | 接CCI总线 |
| AWBARM[1:0] | Output | Write barrier type. | - |
| AWBURSTM[1:0] | Output | Write burst type. | - |
| AWCACHEM[3:0] | Output | Write cache type. | - |
| AWDOMAINM[1:0] | Output | Write shareability domain type. | - |
| AWIDM[4:0] | Output | Write address ID. | - |
| AWLENM[7:0] | Output | Write burst length. | - |
| AWLOCKM | Output | Write lock type. | - |
| AWPROTM[2:0] | Output | Write protection type. | - |
| AWREADYM | Input | Write address ready. | - |
| AWSIZEM[2:0] | Output | Write burst size. | - |
| AWSNOOPM[2:0] | Output | Write snoop request type. | - |
| AWUNIQUEM | Output | 用于WriteBack, WriteClean和WriteEvict传输，指示该写操作为  0 Shared.  1 Unique. | - |
| AWVALIDM | Output | Write address valid. | - |

**W通道**

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 方向 | 描述 | 集成处理 |
| WDATAM[127:0] | Output | Write data | 接CCI总线 |
| WIDM[4:0] | Output | Write data ID？写数据间插？ | - |
| WLASTM | Output | Write data last transfer indication | - |
| WREADYM | Input | Write data ready | - |
| WSTRBM[15:0] | Output | Write byte-lane strobes | - |
| WVALIDM | Output | Write data valid | - |

**B通道**

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 方向 | 描述 | 集成处理 |
| BIDM[4:0] | Input | Write response ID | 接CCI总线 |
| BREADYM | Output | Write response ready | - |
| BRESPM[1:0] | Input | Write response | - |
| BVALIDM | Input | Write response valid | - |

**AR通道**

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 方向 | 描述 | 集成处理 |
| ARADDRM[43:0] | Output | Read address.  The top 4 bits communicate only the ACE virtual address for DVM messages.  The top 4 bits are Read-as-Zero if a DVM message is not being broadcast. | 接CCI总线 |
| ARBARM[1:0] | Output | Read barrier type. | - |
| ARBURSTM[1:0] | Output | Read burst type. | - |
| ARCACHEM[3:0] | Output | Read cache type. | - |
| ARDOMAINM[1:0] | Output | Read shareability domain type. | - |
| ARIDM[5:0] | Output | Read address ID. | - |
| ARLENM[7:0] | Output | Read burst length. | - |
| ARLOCKM | Output | Read lock type. | - |
| ARPROTM[2:0] | Output | Read protection type. | - |
| ARREADYM | Input | Read address ready. | - |
| ARSIZEM[2:0] | Output | Read burst size. | - |
| ARSNOOPM[3:0] | Output | Read snoop request type. | - |
| ARVALIDM | Output | Read address valid. | - |

**R通道**

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 方向 | 描述 | 集成处理 |
| RDATAM[127:0] | Input | Read data | 接CCI总线 |
| RIDM[5:0] | Input | Read data ID | - |
| RLASTM | Input | Read data last transfer indication | - |
| RREADYM | Output | Read data ready | - |
| RRESPM[3:0] | Input | Read data response | - |
| RVALIDM | Input | Read data valid | - |

**AC通道**

Coherency address

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 方向 | 描述 | 集成处理 |
| ACADDRM[43:0] | Input | Snoop address.  The top 4 bits communicate only the ACE virtual address for DVM messages. | 接CCI总线 |
| ACPROTM[2:0] | Input | Snoop protection type. | - |
| ACREADYM | Output | Master ready to accept snoop address. | - |
| ACSNOOPM[3:0] | Input | Snoop request type. | - |
| ACVALIDM | Input | Snoop address valid. | - |

**CR通道**

Coherency response

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 方向 | 描述 | 集成处理 |
| CRREADYM | Input | Slave ready to accept snoop response | 接CCI总线 |
| CRVALIDM | Output | Snoop response | - |
| CRRESPM[4:0] | Output | Snoop response valid | - |

**CD通道**

Coherency data

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 方向 | 描述 | 集成处理 |
| CDDATAM[127:0] | Output | Snoop data | 接CCI总线 |
| CDLASTM | Output | Snoop data last transform | - |
| CDREADYM | Input | Slave ready to accept snoop data | - |
| CDVALIDM | Output | Snoop data valid | - |

**RW ack**

Read and write acknowledge

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 方向 | 描述 | 集成处理 |
| RACKM | Output | Read acknowledge | 接CCI总线 |
| WACKM | Output | Write acknowledge | 接CCI总线 |

## ACP接口

Cortex-A53配置包含ACP接口时才有该部分信号。

All ACP channels must be balanced with respect to CLKIN and timed relative to ACLKENS.

**时钟和配置**

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 方向 | 描述 | 集成处理 |
| ACLKENS | Input | 见时钟部分 | - |
| AINACTS | Input | ACP master不活动且不参与一致性，不可以有outstanding的传输。  当master将该信号置位时，master不可以发任何传输：  0 ACP Master is active.  1 ACP Master is inactive.  注：  当处理器进入L2 WFI低功耗状态时，该信号必须置位。  （接到ACP master， 如果该信号为LOW，处理器会禁止将STANDBYWFIL2置位） | 固结为0，使用ACP接口 |

**AW通道**

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 方向 | 描述 | 集成处理 |
| AWREADYS | Output | Write address ready | ACP\_SHAPER |
| AWVALIDS | Input | Write address valid | - |
| AWIDS[4:0] | Input | Write address ID | - |
| AWADDRS[39:0] | Input | Write address | - |
| AWLENS[7:0] | Input | Write burst length | - |
| AWCACHES[3:0] | Input | Write cache type | - |
| AWUSERS[1:0] | Input | Write attributes:  [0] Inner Shareable.  [1] Outer Shareable. | - |
| AWPROTS[2:0] | Input | Write protection type | - |

**W通道**

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 方向 | 描述 | 集成处理 |
| WREADYS | Output | Write data ready | ACP\_SHAPER |
| WVALIDS | Input | Write data valid | - |
| WDATAS[127:0] | Input | Write data | - |
| WSTRBS[15:0] | Input | Write byte-lane strobes | - |
| WLASTS | Input | Write data last transfer indication | - |

**B通道**

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 方向 | 描述 | 集成处理 |
| BREADYS | Input | Write response ready | ACP\_SHAPER |
| BVALIDS | Output | Write response valid | - |
| BIDS[4:0] | Output | Write response ID | - |
| BRESPS[1:0] | Output | Write response | - |

**AR通道**

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 方向 | 描述 | 集成处理 |
| ARREADYS | Output | Read address ready | ACP\_SHAPER |
| ARVALIDS | Input | Read address valid | - |
| ARIDS[4:0] | Input | Read address ID | - |
| ARADDRS[39:0] | Input | Read address | - |
| ARLENS[7:0] | Input | Read burst length | - |
| ARCACHES[3:0] | Input | Read cache type | - |
| ARUSERS[1:0] | Input | Read attributes:  [0] Inner Shareable.  [1] Outer Shareable. | - |
| ARPROTS[2:0] | Input | Read protection type | - |

**R通道**

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 方向 | 描述 | 集成处理 |
| RREADYS | Input | Read data ready | ACP\_SHAPER |
| RVALIDS | Output | Read data valid | - |
| RIDS[4:0] | Output | Read data ID | - |
| RDATAS[127:0] | Output | Read data | - |
| RRESPS[1:0] | Output | Read response | - |
| RLASTS | Output | Read data last transfer indication | - |

## 外部调试接口

### APB接口

You must balance all APB interface signals with respect to CLKIN and time them relative to PCLKENDBG.

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 方向 | 描述 | 集成处理 |
| PCLKENDBG | Input | 见时钟部分 | - |
| nPRESETDBG | Input | 见复位部分 | - |
| PADDRDBG31 | Input | APB address bus bit[31]:  0 Not an external debugger access.  1 External debugger access.  (接Debug APB 总线) | DAP控制器过来的访问识别控制为1，系统过来的访问识别控制位0； |
| PADDRDBG[21:2] | Input | APB address bus.  (接Debug APB 总线) | 接Debug apb网络 |
| PENABLEDBG | Input | Indicates the second and subsequent cycles of an APB transfer.  (接Debug APB 总线) | - |
| PRDATADBG[31:0] | Output | APB read data.  (接Debug APB 总线) | - |
| PREADYDBG | Output | APB slave ready.  An APB slave can deassert PREADYDBG to extend a transfer by inserting wait states.  (接Debug APB 总线) | - |
| PSELDBG | Input | Debug bus access.  (接Debug APB 总线) | - |
| PSLVERRDBG | Output | APB slave transfer error:  0 No transfer error.  1 Transfer error.  (接Debug APB 总线) | - |
| PWDATADBG[31:0] | Input | APB write data.  (接Debug APB 总线) | - |
| PWRITEDBG | Input | APB read or write signal:  0 Reads from APB.  1 Writes to APB.  (接Debug APB 总线) | - |

对于 PADDRDBG[31]，地址的最高位，表示当前的访问是 internal access，还是 external access。

* internal access，是指处理器执行指令的访问，比如 load/store 去访问， 或者是外部debugger 通过 memory map 的访问。
* external access，是指外部的访问，比如 debugger， external access 比 internal access 有更高的权限。

如果 debugger 使用地址 0x0000\_0000 访问，是 internal access， 此时受限于 software lock 的影响。而使用地址 0x8000\_0000 访问，是 external access，不受限于 software lock。

### 其他

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 方向 | 描述 | 集成处理 |
| DBGROMADDR[39:12] | Input | Debug ROM基地址  指定ROM table物理地址的bits[39:12]  如果地址不能确定，则接低电平。  该信号只在处理器复位期间被采样。  （只能在macrocell处于复位时，修改该值，系统中如果有多个debug ROM，必须将最顶层 ROM的地址设置到该接口） | ？  外部顶层ROM table基地址 |
| DBGROMADDRV | Input | Debug ROM 基地址有效.  如果地址不能确定，则接低电平。  该信号只在处理器复位期间被采样。  （如果debug ROM物理地址不能指定，必须设置：  ----DBGROMADDR[39:12]设置为全0.  ----DBGROMADDRV接LOW） | 固结1，  如果没有系统ROM table。 |
| DBGACK[CN:0] | Output | Debug acknowledge:  0 外部调试请求未确认。  1外部调试请求确认。  （接到debug controller） | 系统控制器可读 |
| EDBGRQ[CN:0] | Input | External debug request:  0 No external debug request.  1 External debug request.  处理将该信号以电平敏感对待，信号要保持到处理器将DBGACK置位。  （系统中所有debug请求源必须使用or逻辑接到该信号上） | 系统控制器控制，默认值未0. |
| nCOMMIRQ[CN:0] | Output | Communications channel receive or transmit interrupt request  0 Request interrupt.  1 No interrupt request.  （低有效，电平敏感）  （DDC的中断请求基于DDC的状态标志被通知。  如果需要使用中断驱动调试通信操作，则将nCOMMIRQ或者COMMRX和COMMTX接到中断控制器。） | GIC-PPI-22 |
| COMMRX[CN:0] | Output | Communications channel receive. Receive portion of Data Transfer Register full flag:  0 Empty.  1 Full. | 悬空 |
| COMMTX[CN:0] | Output | Communication transmit channel. Transmit portion of Data Transfer Register empty flag:  0 Full.  1 Empty. | 悬空 |
| DBGEN[CN:0] | Input | 侵入式（Invasive）调试使能:  0 Not enabled.  1 Enabled.  （如果需要调试功能，该信号接高电平，例如外部调试使用DSTREAM，或自主调试） | 通过efuse rom来控制，调试或测试模式下，支持使能。 |
| NIDEN[CN:0] | Input | 非侵入式（Non-invasive）调试使能:  0 Not enabled.  1 Enabled.  （如果该信号接低电平，则非侵入式调试功能不可用，如PMU和trace） | 通过efuse rom来控制，调试或测试模式下，支持使能。 |
| SPIDEN[CN:0] | Input | 安全特权侵入式（Secure privileged invasive）调试使能：  0 Not enabled.  1 Enabled. | 通过efuse rom来控制，调试或测试模式下，支持使能。 |
| SPNIDEN[CN:0] | Input | 安全特权非侵入式（Secure privileged non-invasive）调试使能：  0 Not enabled.  1 Enabled. | 通过efuse rom来控制，调试或测试模式下，支持使能。 |
| DBGRSTREQ[CN:0] | Output | 见复位部分 | - |
| DBGNOPWRDWN[CN:0] | Output | 见电源管理部分 | - |
| DBGPWRUPREQ[CN:0] | Output | 见电源管理部分 | - |
| DBGPWRDUP[CN:0] | Input | 见电源管理部分 | - |
| DBGL1RSTDISABLE | Input | 见复位部分 | - |

## ATB接口

You must balance all ATB interface signals with respect to CLKIN and time them relative to ATCLKEN.

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 方向 | 描述 | 集成处理 |
| ATCLKEN | Input | 见时钟部分 | - |
| ATREADYMx | Input | ATB device ready | 连接到系统的trace逻辑 |
| AFVALIDMx | Input | FIFO flush request |
| ATDATAMx[31:0] | Output | Data |
| ATVALIDMx | Output | Data valid |
| ATBYTESMx[1:0] | Output | Data size |
| AFREADYMx | Output | FIFO flush finished |
| ATIDMx[6:0] | Output | Trace source ID |

## ETM trace unit其他

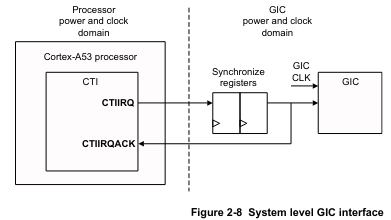
|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 方向 | 描述 | 集成处理 |
| ATCLKEN | Input | 见时钟部分 | - |
| SYNCREQMx | Input | Synchronization request from trace sink  （如果系统的trace sink支持全局同步，则连接到trace逻辑。不需要的话接低电平） | 根据时钟域配该信号，待定 |
| TSVALUEB[63:0] | Input | Timestamp in binary encoding  （系统中如果有timestamp生成器，则将该信号连接到该生成器。如果没有接低电平） | CoreSight timestamp系统 |

## CTI接口

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 方向 | 描述 | 集成处理 |
| CLKIN | Input | 见时钟部分 | - |
| CTICHIN[3:0] | Input | Channel In  （连到系统级CTI或CTM的CTICHOUT） | CTI，系统trigger网络 |
| CTICHOUTACK[3:0] | Input | Channel Out acknowledge  （如果通道接口不需要握手，则接高电平；  如果通道接口需要握手，则接系统级CTI或CTM的CTICHINACK上。） | - |
| CTICHOUT[3:0] | Output | Channel Out  （连到系统级CTI或CTM的CTICHIN） | - |
| CTICHINACK[3:0] | Output | Channel In acknowledge  （如果通道接口不需要握手，则接高电平；  如果通道接口需要握手，则接系统级CTI或CTM的CTICHOUTACK上。） | - |
| CISBYPASS | Input | Channel interface sync bypass  （接高电平旁路CTICHIN和CTICHOUTACK的时钟域同步器。  如果系统级CTI和CTM时钟与CLKIN异步，则需要将该信号接低电平） | - |
| CIHSBYPASS[3:0] | Input | Channel interface H/S bypass  （某bit接高电平旁路对应的CTICHOUT握手）  （例：如果是中断触发，就没有握手；如果是来自另一个CTI/CTIM，就需要握手） | 待定 |
| CTIIRQ[CN:0] | Output | CTI interrupt (active-HIGH)  （高有效，脉冲信号）  （接中断控制器）  （处理见后面说明） | GIC-PPI-24（需要处理后连接到GIC） |
| CTIIRQACK[CN:0] | Input | CTI interrupt acknowledge  （可以与CLKIN异步）  （如果CTIIRQ上不需要握手，则该信号接高电平。接低电平允许软件握手。如果中断控制器使用电平敏感输入，必须使用软件握手。） | 见后述 |

系统集成时连接CTIIRQACK反馈路径见图2-8。反馈路径穿过寄存器，保证A53发出的脉冲持续有效，直到GIC域可见。

CTIIRQACK是一个内部有同步器的输入，不需要在外面加同步器。



（要不要取反接到GIC-PPI？GIC-500 PPI默认低电平敏感，或编程为上升沿触发，不需要取反）

## PMU接口

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 方向 | 描述 | 集成处理 |
| PMUEVENTx[29:0] | Output | PMU event bus  （连接到系统的trace逻辑） | 要接CTI网络吗？  TESTMUX输出；  或者采样到系统控制器，可编程控制采样某些位，并记录第一次采样值？ |
| nPMUIRQ[CN:0] | Output | PMU interrupt request  （低有效，电平敏感） | GIC-PPI-23 |

## DFT和MBIST

### DFT

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 方向 | 描述 | 集成处理 |
| DFTRAMHOLD | Input | Disable the RAM chip select during scan testing  （接测试控制器，不使用的话接低电平） | 接DFT顶层，待定 |
| DFTRSTDISABLE | Input | Disable internal synchronized reset during scan shift  （接测试控制器，不使用的话接低电平） | 接DFT顶层，待定 |
| DFTSE | Input | Scan shift enable, forces on the clock grids during scan shift  （接测试控制器，不使用的话接低电平） | 接DFT顶层，待定 |
| DFTMCPHOLD | Input | Disable Multicycle Paths on RAM interfaces  （接测试控制器，不使用的话接低电平） | 接DFT顶层，待定 |

### MBIST

注意

Cortex-A53 处理器不包括外部MBIST地址和数据接口。 MBIST地址和数据接口在设计中是内部tied-off的，您可以在综合之前将MBIST插入到设计中。将MBIST添加到设计中的过程可以由EDA MBIST工具自动完成。

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 方向 | 描述 | 集成处理 |
| MBISTREQ | Input | MBIST test request  （正常功能模式时，该信号控制为低电平。  MBIST模式时，该信号控制位高电平，然后通过MBIST接口进行MBIST测试） | DFT控制，待定 |
| nMBISTRESET | Input | 见复位部分 | - |