**支持赛灵思嵌入式 RDMA 的网卡 v3.1**

***逻辑核知识产权产品指南***

**Vivado 设计套件**

**PG332 6月 30， 2021**

# 目录

**知识产权事实**

## 第 1 章：概述

**通过设计过程导航内容 . 5 核心概述 . . . . . . . . . . . . . . . . . 5 功能摘要 . 7 申请 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 10 不支持的功能 . 10**

**许可和订购 . 11**

## 第2章 产品规格

**启用 RDMA 的网卡 .. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 12 工作请求/工作队列条目 （WQE） . 13**

**工程竣工 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 14 RDMA 队列 . 14 厄尼克RX路径 . 16 德克萨斯州厄尼克路径**  .**. . . . . . . . . . . . . . . . 21 标准 . 24 表演 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 24 资源利用 24 港口说明 . 25 参数说明 . . . . . . . . . . . 29**

**注册空间 . 30**

## 第 3 章：使用核心进行设计

**一般设计准则 . .. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 47 中断 . 48 打卡 49**

**重 置。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 49**

## 第 4 章：设计流程步骤

**定制和生成核心 . 50 模拟 .. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 51**

**综合与实施 . . . . . 51**

[发送反馈](https://www.xilinx.com/about/feedback/document-feedback.html?docType=Product_Guide&docId=PG332&Title=Xilinx%20Embedded%20RDMA%20Enabled%20NIC%20v3.1&releaseVersion=3.1&docPage=2) **2**

## 第 5 章：示例设计

**设计特征示例 . 53 示例设计限制 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 54 模拟示例设计 . 54**

**示例序列 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 55**

## 第 6 章：ERNIC 软件流程

**ERNIC 全局配置 . . . . 57 QP1 创造 . 58 记忆登记 .. . . . . . . . . . . . . . . . . . . . . . . . . . 58 RC QP的创造 . . 59 WQE SQ 发布 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 60 接收传入的 RDMA 发送消息 .. 60 处理WQE完成情况 60 启用 QP 硬件卸载 . 61 QP删除 . 61**

**QP致命复苏 . . . . . . . . . . . . . . . . . . . . . . . . . 62**

**附录 A：请求 ERNIC 支持问卷**

**附录 B：调试**

**正在调试 IP/系统 . . . . . . 66**

## 附录 C：其他资源和法律声明

**赛灵思资源 . 70 文档导航器和设计中心 70 参考资料 . 71 修订 历史 . 71 请阅读：重要法律声明 . . . . . . . . . . . 73**

## 介绍

**知识产权事实**

赛灵思® ERNIC™（支持嵌入式 RDMA

NIC） IP 是启用 RDMA over Converged Ethernet （RoCE v2） 的 NIC 功能的实现。这款可参数化的软 IP 内核可与各种 Xilinx 软硬 MAC IP 实现配合使用，通过标准以太网提供高吞吐量、低延迟和完全硬件卸载的可靠数据传输解决方案。ERNIC IP 允许同时连接到运行 RoCE v2 流量的多个远程主机。

## 特征

* 支持 RDMA 功能

° RoCE v2

° 数据包重传错误时由IP在硬件中处理。

* 100 Gb/s 线路速率
* 支持可靠连接 （RC） RDMA 传输服务类型
* QP1 支持发送和接收 MAD 数据包
* 用户界面上的 Hardwa重新握手模式，以支持用户逻辑中的硬件 RDMA 应用程序
* 支持传入和传出 RDMA SEND、RDMA READ、RDMA WRITE、RDMA SEND WITH IMM、RDMA WRITE 和 RDMA SEND WITH INVALID消息类型。
* 设计用于支持多达 255 个 RDMA 队列对（3）
* 支持 IPv4 和 IPv6 数据包
* 支持显式拥塞通知 （ECN）
* 支持对 RoCE 和非 RoCE 流量具有不同优先级的优先级流控制。
* 支持内存注册和保护

|  |  |
| --- | --- |
| **LogiCORE™ 知识产权事实表** | |
| **核心细节** | |
| 支持  设备系列（1） | 维萨尔™ ACAP  Kintex UltraScale+™，  Virtex® UltraScale™，  Virtex UltraScale+， Zynq® UltraScale+ |
| 支持的用户  接口 | AXI4-Lite、AXI4 和 AXI4-Stream |
| 资源 | [性能和资源利用率 Web 页e](https://www.xilinx.com/cgi-bin/docs/ndoc?t=ip+ru;d=ernic.html) |
| **提供核心** | |
| 设计文件 | 加密的 RTL |
| 示例设计 | 威瑞洛格 |
| 试验台 | 未提供 |
| 约束文件 | 赛灵思设计约束 （XDC） |
| 模拟  型 | 未提供 |
| 支持  软件驱动器（2） | Linux 内核驱动程序和用户空间库  德玛 |
| **经测试的设计流程**（2） | |
| 设计条目 | Vivado® 设计套件  维凡多 IP 集成商 |
| 模拟 | 有关支持的仿真器，请参见 [Xilinx 设计工具：发行说明指南](https://www.xilinx.com/cgi-bin/docs/rdoc?v=2021.1;t=vivado+release+notes)。 |
| 合成 | 维瓦多合成 |
| **支持** | |
| 发行说明和已知问题 | 不适用 |
| 所有 Vivado 知识产权  变更日志 | 主 Vivado IP 更新日志： [72775](https://www.xilinx.com/support/answers/72775.html) |
| [赛灵思支持网页](https://www.xilinx.com/support) | |

**笔记：**

1. 有关受支持器件的完整列表，请参见 Vivado IP 目录。
2. 有关第三方工具的支持版本，请参阅[Xilinx 设计工具：发行说明指南](https://www.xilinx.com/cgi-bin/docs/rdoc?v=2021.1;t=vivado+release+notes)。
3. 对于 -1 速度等级的设备设计，对于超过 64 QP 的配置，可能会有时序冲突。

[发送反馈](https://www.xilinx.com/about/feedback/document-feedback.html?docType=Product_Guide&docId=PG332&Title=Xilinx%20Embedded%20RDMA%20Enabled%20NIC%20v3.1&releaseVersion=3.1&docPage=4) **4**

产品规格

*第一章*

# 概述

## 按设计过程导航内容

Xilinx® 文档围绕一组标准设计流程进行组织，可帮助您查找当前开发任务的相关内容。本文档涵盖以下设计过程：

• **硬件、**IP 和平台开发：为硬件平台创建 PL IP 模块，创建 PL 内核、子系统功能仿真，并评估 Vivado 时序、资源和电源收敛。还涉及开发用于系统集成的硬件平台。本文档中适用于此设计过程的主题包括：

° 端口说明

° 寄存器空间

° 时钟

° 复位

° 定制和生成核心

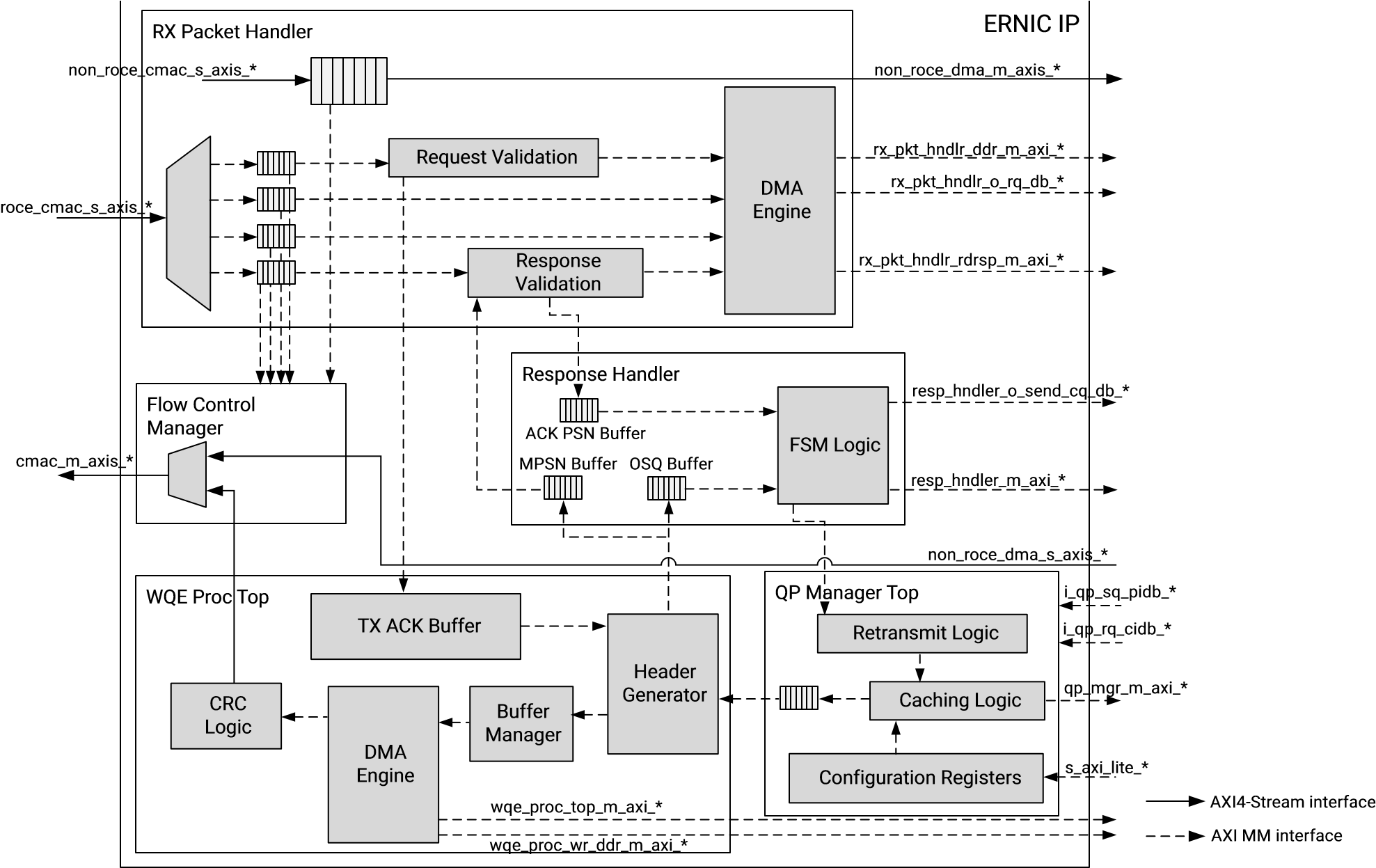
° 设计示例

## 核心概述

本章概述了 ERNIC IP 内核以及应用程序、许可要求和标准一致性的详细信息。ERNIC 是一种通过融合以太网 （RoCE v2） 协议实现 RDMA 的软 IP，用于嵌入式目标或发起方。此实现基于 *InfiniBand 架构规范第* 1 卷、附录 A16 RoCE 和附录 17 RoCE V2 [参考文献 1] 中描述的规范。

图 1-1 显示了 ERNIC 及其与子系统中其他 IP 的连接。

X-参考目标 - 图 1-1



X25376-052621

*图 1-1：* **ERNIC IP 框图**

***注意：*** 连接到 ERNIC 的用户逻辑或目标 IP 称为应用程序，箭头的方向是从主到从。

除 ERNIC IP 外，ERNIC 子系统还包括 Xilinx 以太网 IP、AXI DMA 和 AXI 互连等 IP。在用户应用方面，ERNIC IP 公开了边带接口，以实现高效的门铃交换，而无需通过互连。

每个队列都由一组读取和写入指针标识，称为生产者索引（写入指针）和使用者索引（读取指针）。这些指针的寄存器地址位置在本文档中称为 门铃。门铃交换或门铃响铃表示相应的寄存器位置已更新。

## 功能摘要

ERNIC IP 使用 AXI4 流接口与任何以太网 MAC IP 接口。访问 DDR 或任何其他内存区域对于读取和写入 用于 RDMA 数据包处理的各种数据结构是必需的。此连接是使用多个 AXI4 接口实现的。该 IP 在 512 位内部数据路径上工作，该路径可以完全硬件加速，无需任何软件干预即可进行数据传输。所有可恢复的故障（如由于丢包而导致的重新传输）也完全在硬件中处理。

ERNIC IP 实现了嵌入式 RNIC 功能。因此，与通用 RNIC 相比，仅实现了 RoCE v2 功能的以下子集：

* 支持传入和传出数据包的 RDMA SEND、RDMA READ、RDMA WRITE、RDMA SEND INVALIDATE、RDMA SENDIMME DIATE 和 RDMA WRITE IMMEDIATE 处理。不支持原子操作。
* 最多支持 254 个连接。
* 多达 255 个 RDMA 队列对的可扩展设计。

***注意：***默认 Vivado 策略允许时序传递多达 12 个 7 个队列对。要匹配 255 个队列对的时间，请使用 Vivado 策略 — Performance\_refinePlacement。

* 支持动态内存注册。
* 硬件握手机制，用于与用户应用程序逻辑进行高效门铃交换。

***注意：*** 在握手模式下进行巫术时，上层在该特定 QP 上不应有任何流量。

### 厄尼克模块

ERNIC IP 由本节中介绍的以下主要模块组成。

* QP经理
* WQE 处理器引擎
* RX PKT 交易
* 响应处理程序
* 流量控制管理器

#### QP经理

QP 管理器模块包含所有 QP 的配置，并为处理器提供 AXI4-Lite 接口。它还跨各种 SEND 队列进行仲裁，并缓存 SEND 工作队列条目 （WQE）。然后将这些 WQE 提供给 WQE 进程ssor 模块进行进一步处理。此模块还处理重新传输时的 QP 指针更新。

#### WQE 处理器引擎

WQE 处理器引擎从 QP 管理器模块读取缓存的 WQE，并处理以下任务：

* 验证传入的 WQE 是否存在任何无效的操作码。
* 基于有效负载最大传输单元 （PMTU） 为 RDMA 数据包创建标头，并对内部 DMA 引擎进行编程。
* 它还会触发 DMA 启动传出数据包传输。

WQE 处理器引擎还可用于为传入的 RDMA 发送/写入请求发送传出确认数据包，并为传入的 RDMA 读取请求发送读取响应。

#### RX PKT 交易

RX PKT 处理程序模块接收传入的 RDMA 数据包。在接收 RX PKT 处理程序 （**roce\_cmac\_s\_axis**） 接口之前，应过滤掉非 RDMA 数据包。ERNIC IP 处理以下类型的传入 RoCE v2 数据包：

* RDMA SEND、RDMA WR ITE、RDMA READ和 RDMA READ 的响应数据包（从 ERNIC 发送的请求）
* RDMA SEND with Invalidate、RDMA SEND with imimmediate 和 RDMA WRITE with immediate packets（英语：RDMA SEND with Invalidate）、RDMA SEND with Invalidate
* RDMA 写入/RDMA 发送的确认数据包（从 ERNIC 发送的请求）
* 通信 m 管理（管理数据报） 数据包到 QP1

RX PKT 处理程序模块负责验证传入的数据包。它还会为传入的 RDMA SEND 和 RDMA WRITE 请求触发传出确认数据包，并将通过验证的数据包推送到相应的内存位置。RDMA READ 响应直接定向到目标应用程序。该模块处理传入的 RDMA 读取请求，并将请求转发到 TX 路径。

RX PKT 处理程序模块还解码 RDMA SEND 无效/立即和 RDMA 写入立即数据包。IETH 或 IMMDT 标头中的 32 位数据在单独的 AXI4-Stream 接口上提供。此流接口上为每个条目提供 64 位数据。 下表显示了这 64 位的编码。

*表 1-1：* **无效/即时数据输入结构**

|  |  |  |
| --- | --- | --- |
| **位** | **田** | **描述** |
| 31:0 | ImmDt/R\_KEY | 此字段的内容由“类型”字段定义。此字段包含发送无效时的即时数据或R\_KEY。 |
| 39:32 | 类型 | 此字段指示触发此完成的传入数据包的类型。  8'h01 = 发送无效  8'h02 = 立即发送  8'h03 = WRITE\_IMMEDIATE  8'h04 至 8'hFF = 保留 |
| 47:40 | QPID | 触发此条目的 QP 标识符 |
| 55:48 | RQ PI 指针 | 仅当“类型”字段指示“立即发送”或“发送”表示无效命令时，此字段才有效。此字段指示接收队列指针，其中相应的 SEND 命令位于d。 |
| 63:56 | 保留 | 保留 |

需要实现外部硬件逻辑来处理此接口上提供的此无效/即时数据。例如，AXI DMA IP将流接口转换为AXI存储器映射接口，并将数据写入特定位置并通知 软件或硬件应用程序。

RDMA 流量的传出暂停请求在此模块内处理。当剩余缓冲区位置达到 XON 条件时，它会触发暂停 ON 并在缓冲区指针达到 XOFF 条件时取消断言。

#### 响应处理程序

响应处理程序模块管理未完成的队列。这些队列保存有关发送到远程主机但尚未确认或响应的所有数据包的信息。此外，如果远程主机发送否定确认 （NAK），此模块将触发重新传输。如果此模块在指定时间（超时值）内未收到来自远程主机的响应，则会触发与超时相关的重新传输。

由于传入的 NACK 或内部超时而触发重新传输。触发重传时，ERNIC IP 会处理以下事项：

1. 标识要重新传输的 PSN 和 MSN 值。
2. 回滚该队列对的 PSN 和 MSN 值。
3. 将当前 SQ 指针回滚到需要重新传输的 WQE。
4. ERNIC 在重试的情况下重新传输完整的 WQE。不支持部分 WQE 重传。

#### 流量控制管理器

该模块在RX路径中维护单独的缓冲器，并根据这些缓冲器的可编程完整条件生成PFC控制信号 。同样，在 TX 路径中，它也有两个单独的缓冲区，用于 RoCE 和非 RoCE。断言任何暂停信号时，模块将停止从缓冲区读取。

## 应用

ERNIC IP 可用于需要 通过网络结构可靠传输数据包的各种应用。下面列出了一些这样的应用程序：

* 通过 RoCE V2 进行传感器数据采集和传输
* 通过 RoCE V2 捕获和传输视频/图像
* RoCE V2 上的远程存储节点

## 不支持的功能

ERNIC IP 不支持：

* 传入的原子操作
* 传出原子操作
* 传入/传出 RDMA 发送长度为 0 的数据包
* RDMA 有效负载超过 8 MB 的最大 DMA 长度
* DMA 长度等于或小于 4 字节的传入 RDMA R EAD 请求
* 虚拟地址不在 64 字节边界上的传入 RDMA 读/写请求
* 传入读取响应和写入请求长度不是 64 字节的倍数

## 许可和订购

此赛灵思 LogiCORE ™ IP 模块根据[赛灵思酷睿许可协议](https://www.xilinx.com/cgi-bin/docs/ipdoc?t=turbo+cores+license)的条款提供。该模块作为 Vivado® 设计套件的一部分提供。要完全访问仿真和硬件中的所有核心功能，您必须购买内核许可证。要生成完整的许可，请访问[产品许可网页。](https://www.xilinx.com/getlicense) 评估许可证和硬件超时许可证可能可用于此核心或子系统。请联系您[当地的 Xilinx 销售代表](https://www.xilinx.com/about/contact.html)，了解有关定价和供货情况的信息。

### 许可证检查器

如果 IP 需要许可证密钥，则必须验证密钥。Vivado 设计工具具有多个许可证检查点，用于通过流程对许可 IP 进行门控。如果许可证检查成功，IP 可以继续生成。否则，生成将因错误而停止 。许可证检查点由以下工具强制执行：

* 维瓦多合成
* 维瓦多实施
* write\_bitstream（Tcl 命令）

**重要说明：** *检查点将忽略 IP 许可证级别。测试确认存在有效的许可证。它不检查 IP 许可级别。*

# *第2章* 产品规格

ERNIC IP 提供启用 RoCE v2 的 NIC 的嵌入式实现。RDMA技术允许在标准以太网上更快地移动数据，同时完全卸载 CPU 带宽。ERNIC IP 核带有软件驱动程序，可以移植到任何 Zynq® MPSoC 或 FPGA 设备。这允许 ERNIC IP 独立于任何外部处理器运行。

## 启用 RDMA 的网卡

支持 Xilinx® 嵌入式 RDMA 的 NIC （ERNIC） 控制器可使用以下方法之一与任何用户应用接口：

* 紧密集成的硬件握手机制，为 目标系统提供低延迟、轻量级接口
* OFED 投诉 RDMA 硬件驱动程序和用户空间库提供标准软件 API 以将工作请求发布到 ERNIC

图 2-1 显示了具有多个主机 CPU 和多个本机 NVMe 设备通过 Xilinx ERNIC + NVMEOFABRIC IP 子系统通过网络结构进行通信的端到端系统示例。

X-参考目标 - 图 2-1

赛灵思

SoC

赛灵思

SoC

中央处理器

核酸核酸

中央处理器

核酸核酸

中央处理器

核酸核酸

以太网

核酸核酸

1

x100G

普西

普西

普西

普西

核酸核酸

2

x100G

普西

普西

普西

普西

普西

开关

NVMe

NVMe

NVMe

X19877-120220

*图 2-1：* **赛灵思 ERNIC + NVMe 通过互连**

图 2-1 中所示的主机端启用 RDMA 的网卡 （RNIC） 需要支持 RoCE v2 技术。以下各节描述了 ERNIC 实现中使用的关键数据结构。

## 工作请求/工作队列条目 （WQE）

工作请求用于向 ERNIC IP 提交工作单元。应用程序可以发布到“发送工作队列”的操作包括：

* RDMA 写入
* RDMA 立即写入
* RDMA 发送
* RDMA 立即发送
* RDMA 读取
* RDMA 发送无效

不允许以下工作请求：

* 原子
* 绑定内存窗口
* 本地失效
* 快速寄存器物理磁共振成像

接收队列工作请求无需由应用程序发布，因为 ERNIC 硬件会根据配置的接收队列深度自动重新发布使用的接收缓冲区。

表 2-1 显示了 发送工作请求的结构。每个工作队列条目 （WQE） 的大小为 64 字节。

*表2-1：* **WQE结构**

|  |  |  |  |
| --- | --- | --- | --- |
| **位宽** | **内容** | **尺寸 （B）** | **评论** |
| [15:0] | 里德 | 2 | 工作请求 ID。每个 WQE 的唯一标识符 |
| [31:16] | 保留 | 2 |  |
| [95:32] | 拉德 | 8 | 本地地址。操作的数据缓冲区地址 |
| [127:96] | 长度 | 4 | 转移时长 |
| [135:128] | 操作码 | 1 | 8'h00 -- RDMA 写入  8'h01 -- 带 IMMDT 的RDMA\_WRITE  8'h02 -- RDMA 发送  8'h03 -- RDMA 发送与 IMMDT 一起发送  8'h04 -- RDMA 读取  8'H0C -- RDMA 发送时无效 保留所有其他值。 |
| [159:136] | 保留 | 3 | 3 |
| [223:160] | 雷斯特 | 8 | 远程偏移地址 |

*表 2-1：* **WQE 结构 *（续）***

|  |  |  |  |
| --- | --- | --- | --- |
| **位宽** | **内容** | **尺寸 （B）** | **评论** |
| [255:224] | 拉格 | 4 | 远程标签 |
| [383:256] | 数据数据 | 16 | RDMA 发送数据。如果要发送的数据小于或等于 16B，则此字段用于表示数据。 |
| [511:384] | 保留 | 16 | - |
| [415:384] | IMMDT 数据 | 4 | 要在 ImmDt 标头中发送的即时数据 |
| [511:416] | 保留 | 12 | 保留 |

## 工程竣工

对于在发送队列上发布的每个 WQE，将发布工作完成情况。不会为接收队列 （RQ） 条目发布完成，而是在传入的 RDMA SEND 请求使用新的接收缓冲区时，每个队列对 （QP） 在 RQWPTRDBADDi 指向的地址处按门铃响起。表 2-2 中给出了完成队列条目 （CQE） 的结构。每个 CQE 的大小为 4 个字节。

*表 2-2：* **完成队列条目结构**

|  |  |  |  |
| --- | --- | --- | --- |
| **位宽** | **内容** | **尺寸 （B）** | **评论** |
| [15:0] | 里德 | 2 | 工作请求 ID。每个 WQE 的唯一标识符 |
| [23:16] | 操作码 | 1 | * 8'd0 = RDMA 写入 * 8'd1 = RDMA 立即写入 * 8'd2 = RDMA 发送 * 8'd3 = RDMA 立即发送 * 8'd4 = RDMA 读取 * 8'd5 - 8'd11 = 保留 * 8'd12 = RDMA 发送无效 * 8'd13 - 8'd255 = 保留 |
| [31:24] | 错误标志 | 1 | 错误标志。设置为 8'b1，以防 QP 进入致命状态 |

## RDMA 队列

ERNIC IP 实现 RDMA 队列，如接收队列 （RQ）、发送队列 （SQ） 和完成队列 （CQ）。这些队列称为队列对或 QP。 SQ 包含用户应用程序发布的发送 WQE。

X-参考目标 - 图 2-2

发送队列 接收队列

sq\_ba rq\_ba

sq\_depth

rq\_ci\_db

待处理的 RQE

stat\_rq\_pidb/

rq\_wrptr\_db

rq\_depth

sq\_pidb

待定的WQEs

stat\_cq\_head/

sq\_cmpl\_db

已处理 WQE，但

等待响应

stat\_curr\_sqptr\_proc

完成队列

sq\_pidb

待定的WQEs

stat\_cq\_head/

sq\_depth

cq\_ba

X19878-102117

*图 2-2* **：RDMA 队列**

RQ 容纳传入的 RDMA SEND 数据包。完成队列通知用户应用程序已完成的 SEND WQE。这些队列中的每一个都作为循环缓冲区实现。各种门铃和写入指针寄存器定义了这些队列的当前状态。图 2-2 显示了不同的队列和定义队列状态的变量/寄存器。突出显示的变量由 ERNIC IP 间接访问。寄存器 CQDBADDi 指向**sq\_cmpl\_db**的地址，RQWPTRDBADDi 寄存器指向rq\_wrptr\_db的地址。

这些队列在 ERNIC IP 之外的内存区域中实现。ERNIC 通过各种 AXI 主接口访问 IP。有关 ERNIC 内存要求的详细信息，请参阅表 3-2。

接下来的几节简要概述了 ERNIC IP 的传入 （RX） 和传出 （TX） 数据流。

## ERNIC RX 路径

ERNIC RX 路径通过 AXI4-Stream 接口从 MAC 获取数据包数据。验证所有传入数据包，并将 所有未通过数据包验证的数据包标头发送到错误缓冲区（由 ERRBUFBA[31：0] 指定的基址），如果值为

XRNICCONF[5] 设置为 1。标头由 ERNIC RX 数据包处理程序模块以错误综合症为前缀，如选项卡le 2-3 所示。这些缓冲区为传入数据包错误提供有用的调试信息。

*表 2-3：* **数据包验证错误综合症**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **错误**  **症候群**  **位索引** | **与 相关的错误** | **优先** | **描述和影响** | **冲击** |
| 0 | 麦克 | 4 | 在  以太网报头与 ERNIC 不匹配  已配置的 MAC 地址 | 丢弃的数据包 |
| 1 | 保留 |  |  |  |
| 2 | IPv4/IPv6 | 5 | IP 版本不符合 ERNIC 配置 | 丢弃的数据包 |
| 3 | IPv4 | 7 | IPv4 标头长度不等于 20 字节 | 丢弃的数据包 |
| 4 | 保留 |  |  |  |
| 5 | IPv4 | 8 | IPv4 中的标志位不是 3'b010 | 丢弃的数据包 |
| 6 | IPv4 | 8 | IPv4 中的片段偏移量不为 0 | 丢弃的数据包 |
| 7 | 保留 |  |  |  |
| 8 | IPv4/IPv6 | 8 | IPv4 标头中的 IPv4 目标地址与配置的 ERNIC IPv4 地址不匹配 | 丢弃的数据包 |
| 9 | IPv4 | 6 | IPv4 标头校验和错误。 | 丢弃的数据包 |
| 10 | IPv4/IPv6 | 8 | IPv4 总长度字段值不在范围内。  IPv6 有效负载长度字段值不在范围内 | 丢弃的数据包 |
| 11 | 保留 |  |  |  |
| 12 | UDP | 9 | UDP 长度字段与 IPv4 总长度或 IPv6 有效负载长度不一致 | 丢弃的数据包 |
| 13 | 乙基斯 | 10 | BTH版本不是4'b0000 | 丢弃的数据包 |
| 14 | 乙基斯 | 10 | 不支持 BTH 中的 QP 目标地址 | 丢弃的数据包 |
| 15 | 乙基斯 | 11 | 在 BTH 的目标地址中指定的 QP 未配置或未启用 | 丢弃的数据包 |
| 16 | 乙基斯 | 14 | 操作码顺序不正确 | NAK 无效发送，QP 移至致命状态 |

*表 2-3：* **数据包验证错误综合症 *（续）***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **错误**  **症候群**  **位索引** | **与 相关的错误** | **优先** | **描述和影响** | **冲击** |
| 17 | 乙基斯 | 10 | 从远程 QP 接收到不支持或保留的操作码请求 | NAk-发送无效，QP移至致命状态 |
| 18 | 乙基斯 | 14 | 对于收到的第一个和中间请求/响应，BTH Pad 位不是 2'b00 | 丢弃的数据包 |
| 19 | 传输层 | 14 | 传输层有效负载与为该 QP 配置的 PMTU 不一致 | 丢弃的数据包 |
| 20 | 内部错误 | 13 | 请求队列已满，数据包被丢弃 | RNR-NAK 已发送 |
| 21 | 传输层 | 12 | 传入请求 PSN 顺序不正确 | NAK 序列错误 |
| 22 | 传输层 | 14 | AETH 综合征畸形。 | 丢弃的数据包 |
| 23 | 传输层 | 14 | 收到的响应错误或 NAK 无效响应 | QP进入致命状态 |
| 24-26 | 保留 |  |  |  |
| 27 | IPv4/IPv6 | 5 | IP4（6） 报头中的 IP4（6） 源地址与 QP 配置的远程节点源地址不匹配 | 丢弃的数据包 |
| 28 | 麦克 | 4 | 以太网标头中的 MAC 源地址与 QP 配置的远程节点 MAC 地址不匹配 | 丢弃的数据包 |
| 29 | 保留 |  |  |  |
| 30 | 链路层 | 2 | 红十字国际委员会的错误 | 丢弃的数据包 |
| 31 | 链路层 | 1 | 食品接触物质错误 | 丢弃的数据包 |

大多数数据包验证错误完全由硬件处理，不需要软件干预。但是，如果传入数据包导致 QP 进入致命状态，则需要软件干预来处理错误并启动断开连接。此类错误可用于由 IPKTERRQBA、IPKTERRQSZ 和 IPKTERRQWPTR 寄存器定义的传入数据包错误状态缓冲区中的软件。每个错误状态缓冲区条目的宽度为 64 位。错误状态的格式如图 2-3 所示。

X-参考目标 - 图 2-3



*图 2-3* **：错误状态格式**

表2-4中显示了致命表解码。

*表 2-4：* **致命代码的解码**

|  |  |  |
| --- | --- | --- |
| **致命代码** | **描述** | **本地/远程错误** |
| 5'b00001 | 操作码顺序检查失败 | 本地检测到的错误 |
| 5'b00010 | 请求数据包长度与 PMTU 配置的不符或焊盘计数检查失败 | 本地检测到的错误 |
| 5'b00011 | 操作码序列错误和数据包长度或焊盘计数错误同时发生 | 本地检测到的错误 |
| 5'b00100 | 收到不受支持的请求操作码，但具有正确的 PSN | 本地检测到的错误 |
| 5'b00101 | QP由于WQE处理器而致命 | 本地检测到的错误 |
| 5'b00110 | QP由于响应处理程序而致命 | 本地检测到的错误 |
| 5'b10010 | 写入数据包长度错误或填充计数检查失败 | 本地检测到的错误 |
| 5'b10001 | 写入操作码序列检查失败 | 本地检测到的错误 |
| 5'B10011 | 上述两个错误同时发生 | 本地检测到的错误 |
| 5'b10101 | R 键检查失败或访问权限检查失败 | 本地检测到的错误 |
| 5'110110 | RETH DMA 长度检查失败 | 本地检测到的错误 |
| 5'b10111 | 上述两个错误同时发生 | 本地检测到的错误 |
| 5'b10000 | 读取请求资源已满 | 本地检测到的错误 |
| 5'b01100 | ACK 响应操作码不正确 | 本地检测到的错误 |
| 5'b01101 | RNR-NAK 向下计数器已过期 | 本地检测到的错误 |
| 5'b01010 | 收到 NAK 无效/RAE/ROE 响应 | 远程错误 |
| 5'b11111 | 读取响应上次数据包错误 | 本地检测到的错误 |
| 5'b11001 | 不良 AETH 综合征 | 本地检测到的错误 |
| 5'b11110 | 读取响应中间长度检查失败 | 本地检测到的错误 |
| 5'B11100 | 最后读取响应，而不是仅读取响应 | 本地检测到的错误 |
| 5'b11011 | 错误响应错误 | 本地检测到的错误 |
| 5'b01001 | RNR 计数器已过期 | 本地检测到的错误 |
| 5'b11010 | 仅读取响应长度检查失败 | 本地检测到的错误 |

传入的 RDMA 发送/写入/读取请求应在 RX 端。所有其他类型的数据包都是传出请求的响应数据包。传入 RDMA SEND 请求的数据流如图 2-4 所示。箭头的方向显示达塔的流动。

在连接的 QP 上收到有效的 RDMA SEND 传入数据包时，数据包内容（不带标头）将推送到相关 QP 的 RX 缓冲区中。ERNIC 会响铃 RQ 生产者索引门铃 （RQPI DB），以指示新数据包可用，无论是使用边带接口还是通过 AXI 接口。这取决于 QPCONFi 的配置[4]。此时，还会向远程主机发布确认。用户应用程序可以通过 响铃 RQ 使用者索引门铃 （RQCI DB） 来通知 ERNIC 已使用新数据包。收到此门铃后，相应的 RX 缓冲区可用于新的传入数据包。

X-参考目标 - 图 2-4

引发剂 RNIC

厄尼克

用户申请

RDMA 发送

已发送数据包

显式或

已发送的合并 ACK

数据包张贴在

RQ buffer， RQPli

门铃响起

边带 I/F，

STATRQPIDBi

更新

X19879-120220

*图 2-4：* **RDMA 发送 RX 数据流**

对于任何 RDMA 设备上的传入 RDMA 读/写请求支持，以及向网络中的其他 RDMA 设备发送读/写请求，设备应知道目标 RDMA 客户端的地址、长度和 RKEY。为此，应用程序应实现自己的协议，以便在任何数据传输之前交换地址和 RKEY 值。图 2-5 和图 2-6 中的示例显示了交换 RDMA 读/写事务所需详细信息的附加RDMA\_SEND操作。 交换详细信息时，RNIC 将向 ERNIC 发送 RDMA 读/写数据包，ERNIC 将使用 ACK 或 RDMA 读取响应进行响应。 图 2-5 显示了 RDMA READ 事务的数据流。 图 2-6 显示了 RDMA WRITE 事务处理的数据流。

X-参考目标 - 图 2-5

厄尼克

核酸核酸

用户申请

王权局张贴

RDMA READ

请求

RDMA 发送方式

胶囊含有

地址信息

WQE发布在

(

目标）。

获取的数据

从本地

缓冲区

单或

多个 RDMA

读取响应

包

根据

DMA 长度。

X24862-120220

*图 2-5：* **RDMA 读取请求 RX 数据流**

X-参考目标 - 图 2-6

聚丙烯

RDMA 写入

请求

显式或

已发送的合并 ACK

写入写入数据

成相应的

记忆

王权局张贴

RDMA 发送方式

胶囊含有

地址信息

(

WQE发布在

目标）。

X24863-121020

*图 2-6：* **RDMA 写入请求 RX 数据流**

## 厄尼克德克萨斯路径

TX 路径由传入 RDMA 读取、RDMA 写入事务和传入 RDMA 发送/写入请求的 ACK 数据包以及传入 RDMA 读取请求的响应组成。根据 SQPIi 门铃，将处理发送工作队列请求。DMA 模块配置为所有传出事务的数据传输。RDMA 写入/发送的 TX 数据流如图 2-7 所示。

用户应用程序请求 ERNIC IP 传输 RDMA WRITE/SEND/READ packet，方法是在特定 QP 的 SEND 队列上发布 WQE 并响铃相应的 SQ 生产者索引门铃 （SQPI DB）。ERNIC 处理 WQE，并根据 WQE 中提供的信息为 RDMA 写入/发送命令提取数据。此数据 以及相关标头在链接上推送。一旦收到来自远程主机的确认，ERNIC 就会根据 QPCONFi[5] 的配置发布 CQE，并通过边带接口发布完成计数，从而通知用户应用程序 WQE 的成功完成 。相应 QP 的 CQHEAD 也会更新。对于 RDMA 读取 请求，RX 数据包处理程序会提示 TX 路径。对于此通信，每个 QP 都有一个未完成的读取请求队列。队列的深度由传入请求资源的参数确定。当未完成的读取请求队列已满时，将指示到 RX 路径。对 QP 的任何进一步请求都将导致 NAK 无效，并且 QP 将进入致命状态。

***注意：*** 箭头的方向显示数据流。

X-参考目标 - 图 2-7

用户申请

核酸核酸

厄尼克

德玛

发送/写入

已发送的数据包

显式或

合并的 ACK

CQ条目已发布，

完成计数

发送方式

边带通道

CQHEADi 已更新

WQE发布，

SQPli DB 振动

RDMA 写入数据

由 ERNIC 从

本地缓冲区

X19881-120220

*图 2-7：* **RDMA 发送/写入数据流**

图 2-8 显示了从 ERNIC 到远程主机的 RDMA READ 请求流。

X-参考目标 - 图 2-8

核酸核酸

厄尼克

目标知识产权

RDMA READ

已发送数据包

读

回应

收到

CQ条目已发布，

完成计数

发送方式

边带通道

CQHEADi 已更新

WQE发布，

SQPli DB 振动

已推送读取响应

到本地缓冲区

X19882-120220

*图 2-8：* **RDMA 读取数据流**

图 2-8 显示了从 ERNIC 向远程主机发出 RDMA READ 请求的流程。用户应用程序请求 ERNIC IP 将 RDMA READ 请求传输到远程主机，方法是在发送队列上发布特定 QP 的 WQE 并按响相应的 SQ 生产者索引门铃 （SQPI DB）。ERNIC 处理 WQE 并创建具有相关标头的请求数据包，并将其推送到链路上。从远程主机接收到响应数据包后，ERNIC 将数据（在删除头 ers 之后）写入WQE 中提供的本地缓冲区地址。然后，它根据 QPCONFi[5] 的配置发布 CQE，并通过边带间发布完成计数，通知用户应用程序 WQE 的成功完成。相应 QP 的 CQHEAD 也会更新。

## 标准

此实现基于 *InfiniBand 架构规范第* 1 卷、附录 A16 RoCE 和附录 17 RoCE V2 [参考文献 1] 中描述的标准和规范。

## 性能

ERNIC IP 设计的内部数据路径吞吐量高达 100 Gb/s，频率为 200 MHz。

有关资源利用率的更多详细信息，请参阅 [性能和资源利用率。](https://www.xilinx.com/cgi-bin/docs/ndoc?t=ip+ru;d=ernic.html)

## 资源利用率

本节总结了ERNIC IP 中 V 个模块的估计最大性能。数据按设备系列划分为一个表。每行描述一个测试用例。这些列分为测试参数和结果。测试参数包括零件信息和特定于内核的配置参数。未列出的任何配置参数都有其默认值;任何具有空白值的参数都将由 IP 核自动禁用或设置。有关 GUI 参数和用户参数映射的列表，请参阅此 IP 核的产品指南 。

* 资源使用数字取自使用 Vivado 设计套件中的上下文外流程在实施结束时发布的利用率报告。
* 上下文外 IP 约束包括 HD。根据需要CLK\_SRC属性， 以确保正确的保持时序收敛。使用以下 Tcl 命令启用这些属性： **set\_param ips.includeClockLocationConstraint true**
* 用于时钟输入的频率针对每个测试用例进行了说明。
* LUT 编号不包括通过s pack 直通使用的 LUT，但包括用作内存的 LUT。
* 使用默认的 Vivado 设计套件 2018.1 设置。您可以使用不同的设置来改进这些数字。但是，由于周围的电路会影响放置和时序，因此这些数字可能不会在较大的设计中重复出现。

有关资源利用率的更多说明，请参阅 [性能和资源利用率。](https://www.xilinx.com/cgi-bin/docs/ndoc?t=ip+ru;d=ernic.html)

## 端口说明

表 2-5 表描述了端口及其接口定义

*表 2-5：* **ERNIC IP 端口**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **名字** | **I/O** | **宽度** | **时钟**  **域** | **描述** |
| **时钟和重置** | | | | |
| m\_axi\_aclk | 我 | 1 | AXI4 | AXI4 时钟 |
| m\_axi\_aresetn | 我 | 1 |  | AXI4 复位（低电平有效） |
| s\_axi\_lite\_aclk | 我 | 1 | AXI4 | AXI4 精简版时钟 |
| s\_axi\_lite\_aresetn | 我 | 1 |  | AXI4 精简版复位（低电平有效） |
| system\_resetn | 或 | 1 |  | 系统复位 |
| cmac\_rx\_clk | 我 | 1 |  | 来自 CMAC 的 RX 用户时钟输出 |
| cmac\_rx\_rst | 我 | 1 |  | 来自 CMAC 的用户逻辑的 RX 复位 |
| cmac\_tx\_clk | 我 | 1 |  | 来自 CMAC 的 TX 用户时钟输出 |
| cmac\_tx\_rst | 我 | 1 |  | 来自 CMAC 的用户逻辑的 TX 重置 |
| **响应处理程序 AXI4 主接口** | | | | |
| resp\_hndler\_m\_axi\_\* | I/O |  | AXI4 | 响应处理程序使用此接口在 DDR 中存在的完成队列中写入完成。有关 AXI4 信号的更多信息，请参见 Vivado 设计套件附录 A：AXI 参考指南 （UG1037） [参考文献 8]。 |
| **RX 数据包处理程序 AXI4 DDR 主接口** | | | | |
| rx\_pkt\_hndler\_ddr\_m\_axi\* | I/O |  | AXI4 | RX 数据包处理程序模块使用此接口将数据从传入的 MAD、SEND 和 RDMA 写入数据包推送到 DDR 中的 RQ 缓冲区。仅支持 64B 对齐事务。参见 *Vivado 设计套件附录 A：AXI 参考指南* （UG1037）  [参考文献 8] 有关 AXI4 信号的更多信息。 |
| **RX 数据包处理程序 AXI4 主读取响应接口** | | | | |
| rx\_pkt\_hndler\_rdrsp\_m\_axi\* | I/O |  | AXI4 | RX 数据包处理程序使用此接口将读取响应中接收的数据写入 DDR。有关 AXI4 信号的更多信息，请参见 *Vivado 设计套件附录 A：AXI 参考指南* （UG1037） [参考文献 8]。 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **名字** | **I/O** | **宽度** | **时钟**  **域** | **描述** |
| **AXI4 流从属接口，用于传入 RoCE 流量** | | | | |
| roce\_cmac\_s\_ axis\_\* | I/O |  | AXI4-  流 | 此接口提供从网络接口到 IP 的 RoCE 数据包。有关 AXI4 信号的更多信息，请参见 *Vivado 设计套件附录 A：AXI 参考指南* （UG1037） [参考文献 8]。 |
| **用于传入非 RoCE 流量的 AXI4 流从属接口** | | | | |
| non\_roce\_cmac\_s\_ axis\_\* | I/O |  | AXI4-  流 | 此接口提供从网络接口到 IP 的非 RoCE 数据包。有关 AXI4 信号的更多信息，请参见 *Vivado 设计套件附录 A：AXI 参考指南* （UG1037） [参考文献 8]。 |
| **来自 DMA 模块的非 RoCE AXI4 流接口** | | | | |
| non\_roce\_dma\_s\_axis\_\* | I/O |  | AXI4-  流 | 从 DMA 模块到 IP 的传入非 RDMA 路径。有关 AXI4 信号的更多信息，请参见 *Vivado 设计套件附录 A：AXI 参考指南* （UG1037） [参考文献 8]。 |
| **到 DMA 模块的非 RoCE AXI4 流接口** | | | | |
| non\_roce\_dma\_m\_axis\_\* | I/O |  | AXI4-  流 | AXI4 从 IP 到 DMA 模块的传出非 RDMA 路径。有关 AXI4 信号的更多信息，请参见 *Vivado 设计套件附录 A：AXI 参考指南* （UG1037） [参考文献 8]。 |
| **CMAC AXI4 流接口** | | | | |
| cmac\_m\_axis\_\* | I/O |  | cmac\_tx \_clk | 此接口用于将 RoCE 和非 RoCE 数据包从 ERNIC 发送到 CMAC。 |
| **WQE 处理器 AXI 主接口** | | | | |
| wqe\_proc\_top\_m\_axi\_\* | I/O |  | AXI4 | WQE 处理器引擎使用此接口从本地缓冲区读取传出 RDMA 发送/写入和读取响应中发送的数据。参见 *Vivado 设计套件附录 A：AXI 参考指南* （UG1037）  [参考文献 8] 有关 AXI4 信号的更多信息。 |
| **WQE 处理器 AXI4 DDR 主接口** | | | | |
| wqe\_proc\_wr\_ddr\_m\_axi\_\* | I/O |  | AXI4 | WQE 处理器使用此接口将传出写入数据包的数据写入写入重试缓冲区。参见 *Vivado 设计套件附录 A：AXI 参考指南* （UG1037）  [参考文献 8] 有关 AXI4 信号的更多信息。 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **名字** | **I/O** | **宽度** | **时钟**  **域** | **描述** |
| **AXI4-Lite 从机接口，用于寄存器编程** | | | | |
| s\_axi\_lite\_\* | I/O |  | AXI4Lite | 处理器使用此接口来配置 ERNIC 寄存器。有关 AXI4 信号的更多信息，请参见 *Vivado 设计套件附录 A：AXI 参考指南* （UG1037） [参考文献 8]。 |
| **QP 管理器 AXI4 主接口** | | | | |
| qp\_mgr\_m\_axi\_\* | I/O |  | AXI4 | QP 管理器使用此接口从 DDR 获取发送队列 WQE。看  *Vivado 设计套件附录 A：AXI 参考指南* （UG1037） [参考文献 8] 了解有关 AXI4 信号的更多信息。 |
| **使数据无效或立即生效 AXI4 流接口** | | | | |
| ieth\_immdt\_m\_axis\_\* | I/O |  | AXI4S | 此接口用于向外部硬件逻辑提供 IETH 或 Immdt 标头以及 32 位附加数据。此信息的使用留给用户应用程序。 |
| **用于 RQ 门铃的硬件握手端口** | | | | |
| rx\_pkt\_hndler\_o\_rq\_db\_data | 或 | 32 | AXI4 | 来自 ERNIC 的 RDMA-SEND 生产者索引门铃值 |
| rx\_pkt\_hndler\_o\_rq\_db\_addr | 或 | 10 | AXI4 | RDMA-SEND生产者索引门铃地址（每个QP4个字节;127个QP） |
| rx\_pkt\_hndler\_o\_rq\_db\_data\_valid | 或 | 1 | AXI4 | RDMA-SEND生产者索引门铃有效。当此信号被断言为高时，rx\_pkt\_hndler\_o\_rq\_db\_addr和rx\_pkt\_hndler\_o\_rq\_db\_data有效。  直到不对rx\_pkt\_hndler\_i\_rq\_db\_rdy进行采样  高电平时，此信号保持置位，rx\_pkt\_hndler\_o\_rq\_db\_addr 和 rx\_pkt\_hndler\_o\_rq\_db\_data 信号将保持相同的值。 |
| rx\_pkt\_hndler\_i\_rq\_db\_rdy | 我 | 1 | AXI4 | 准备好从目标信令数据和地址被接受 |
| **用于CQ门铃的硬件握手端口** | | | | |
| resp\_hndler\_o\_send\_cq\_db\_cnt | 或 | 32 | AXI4 | 从 ERNIC 发送 WQE 完成队列门铃计数 |
| resp\_hndler\_o\_send\_cq\_db\_addr | 或 | 10 | AXI4 | 发送 WQE 完成队列门铃地址（每个 QP 4 个字节;对于 127 个 QP） |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **名字** | **I/O** | **宽度** | **时钟**  **域** | **描述** |
| resp\_hndler\_o\_send\_cq\_db\_cnt\_valid | 或 | 1 | AXI4 | 发送 WQE 完成门铃有效。什么时候  此信号断言为高，resp\_hndler\_o\_send\_cq\_db\_addr和resp\_hndler\_o\_send\_cq\_db\_cnt有效。在resp\_hndler\_i\_send\_cq\_db\_rdy未采样为高电平之前，此信号保持置位状态，并且resp\_hndler\_o\_send\_cq\_db\_addr和resp\_hndler\_o\_send\_cq\_db\_cnt 信号将保持相同的值。 |
| resp\_hndler\_i\_send\_cq\_db\_rdy | 我 | 1 | AXI4 | 发送 WQE 完成门铃准备好。当目标应用程序接受当前门铃事务时，就绪信号应变为高电平。 |
| **用于 SQ PI 门铃的硬件握手端口** | | | | |
| i\_qp\_sq\_pidb\_hndshk | 我 | 16 | AXI4 | 从目标应用程序发送 WQE 生产者指数门铃值 |
| i\_qp\_sq\_pidb\_wr\_addr\_hndshk | 我 | 32 | AXI4 | 发送 WQE 生产者索引门铃地址 |
| i\_qp\_sq\_pidb\_wr\_valid\_hndshk | 我 | 1 | AXI4 | 发送 WQE 生产者索引门铃有效。目标应用程序发布 WQE 后，应使用有效的i\_qp\_sq\_pidb\_wr\_addr\_hndshk和i\_qp\_sq\_pidb\_hndshk断言此信号 High。目标应用程序应保持此信号置位，并保持i\_qp\_sq\_pidb\_wr\_addr\_hndshk和i\_qp\_sq\_pidb\_hndshk信号，直到其采样o\_qp\_sq\_pidb\_wr\_rdy为高电平。 |
| o\_qp\_sq\_pidb\_wr\_rdy | 或 | 1 | AXI4 | 发送 WQE 生产者索引门铃准备好。当 ERNIC 接受门铃值时，就绪信号断言为高电平。 |
| **用于 RQ CI 门铃的硬件握手端口** | | | | |
| i\_qp\_rq\_cidb\_hndshk | 我 | 16 | AXI4 | 来自目标应用程序的 RDMA-SEND 消费者指数门铃值 |
| i\_qp\_rq\_cidb\_wr\_addr\_hndshk | 我 | 32 | AXI4 | RDMA-SEND消费者索引门铃寄存器地址 |
| i\_qp\_rq\_cidb\_wr\_valid\_hndshk | 我 | 1 | AXI4 | RDMA-SEND消费者指数门铃有效。目标应用程序处理传入的 RDMA-SEND 命令后，应断言此信号为高，并具有有效  i\_qp\_rq\_cidb\_wr\_addr\_hndshk和i\_qp\_rq\_cidb\_hndshk。目标应用程序应保持此信号置言d 并保持i\_qp\_rq\_cidb\_wr\_addr\_hndshk和i\_qp\_rq\_cidb\_hndshk信号，直到它采样o\_qp\_rq\_cidb\_wr\_rdy为高电平。 |
| o\_qp\_rq\_cidb\_wr\_rdy | 或 | 1 | AXI4 | RDMA SEND消费者指数门铃值被ERNIC接受 |
| **名字** | **I/O** | **宽度** | **时钟**  **域** | **描述** |
|  | **优先流量控制端口** | | | |
| stat\_rx\_pause\_req[8：0] | 我 | 9 | cmac\_rx \_clk | 暂停请求信号。CMAC IP 在有效量子周期内断言此信号。 |
| ctl\_tx\_pause\_req[8：0] | 或 | 9 | cmac\_tx \_clk | 当缓冲区阈值介于 XON 和 XOFF 之间时，将断言此总线。优先级位通过优先级寄存器设置。 |
| ctl\_tx\_resend\_pause | 或 | 1 | cmac\_tx \_clk | 此信号硬连线为 0，可以连接到 CMAC IP 信号。 |
|  | **中断** | | | |
| rnic\_intr | 或 | 1 | AXI4 | 当寄存器 INTEN 中的任何一个中断发生时，将设置此位。 |
|  | **调试计数器使能信号** | | | |
| o\_global\_dbg\_cnt\_en | 或 | 1 | AXI4 | 启用调试信号 |
| o\_global\_dbg\_cnt\_clr | 或 | 1 | AXI4 | 清除调试计数器 |

## 参数说明

由于 ERNIC 控制器设计中的许多功能都是使用参数控制的，因此控制器实现可以仅使用所需功能所需的资源进行独特定制。这种方法还可以以最低的资源使用率实现最佳性能。

参数说明和默认值如表2-6所示。

*表 2-6：* **ERNIC 参数**

|  |  |  |
| --- | --- | --- |
| **参数名称** | **描述** | **默认值** |
| C\_NUM\_QP | 队列对的最大数量。最小队列对数为 8，最大队列对数为 256 | 8 |
| C\_M\_AXI\_ADDR\_WIDTH | AXI 主地址宽度（支持的值为 32 和 64） | 32 |
| C\_M\_AXI\_ID\_WIDTH | AXI 主 ID 宽度 | 1 |
| C\_MAX\_WR\_RETRY\_DATA\_BUF\_DEPTH | 重新传输时 RDMA 写入的最大数据缓冲区数。每个 RDMA 数据缓冲区为 4 | 512 |
| C\_EN\_WR\_RETRY\_DATA\_BUF | 启用在 DDR 中存储 RDMA 写入数据以进行重新传输 | 1 |
| C\_EN\_INITIATOR\_LITE | 启用传入的 RDMA 读取和 RDMA 写入数据包 | 0 |
| C\_EN\_DEBUG\_PORTS | 启用调试端口以进行调试 | 0 |

## 寄存器空间

所有 ERNIC 寄存器都与 AXI4-Lite 域同步。以下寄存器表中未指定的任何位都被视为保留位，并在读取时将值返回为 0。控制寄存器的上电复位值为0，除非在定义中指定。除非另有说明，否则应始终使用 0 编写保留位置。下表中仅列出了地址偏移，基址由 AXI 互连在系统级别配置。保护域 ta ble 的内容用于标头验证，如下图所示。

X-参考目标 - 图 2-9

RETH 虚拟地址 LSB

RETH 虚拟地址 MSB

局部放电指数

RETH DMA 长度

PD 编号

虚拟地址 LSB

虚拟地址 MSB

物理地址 LSB

物理地址 MSB

保留

DMA 长度

回复

R\_Key

访问

火柴

R\_KEY

缓冲区

范围

验证

检查

匹配 PD

数

PD 编号

回复

31

23

0

23

31

31

0

7

个人信息信息

**步骤 2**

**步骤 1**

每个 QP 配置中的 PD 编号

保护域表

瑞斯

使用PD索引，这是一部分

的瑞斯·

获取 PD 信息

从 PD 表

31

0

31

0

7

X21690-121020

16

回复

DMA 长度 MSB

*图 2-9：* **PDT 表实现**

**ERNIC 寄存器详细信息**

|  |  |  |  |
| --- | --- | --- | --- |
| **地址** | **访问** | **寄存器名称** | **详** |
| 0x00 + （（i） x 0x0100）  其中 i=0 到 255 | 乌尔曼 | 保护域号码 （PDPDNUM） | 保护域表中的保护域编号  宽度 = 24  默认值 = 0 |
| 0x04 + （（i） x 0x0100） | 乌尔曼 | 虚拟地址 LSB （VIRTADDRLSB） | 此寄存器提供缓冲区基址 LSB 的虚拟添加。  宽度 = 32  默认值 = 0 |
| 0x08 + （（i） x 0x0100） | 乌尔曼 | 虚拟地址 MSB （VIRTADDRMSB） | 此寄存器提供缓冲区基址 MSB 的虚拟添加。  宽度 = 32  默认值 = 0 |
| 0x0C + （（i） x 0x0100） |  | 缓冲区基址  低音水平  （BUFBASEADDRLSB） | 此寄存器提供写/读缓冲区基址 MSB。  宽度 = 32  默认值 = 0 |
| 0x10 + （（i） x 0x0100） | 乌尔曼 | 缓冲区基址  MSB  （BUFBASEADDRMSB） | 此寄存器提供写/读缓冲区基址 LSB。  宽度 = 32  默认值 = 0 |
| 0x14 + （（i） x 0x0100） | 乌尔曼 | 缓冲液R\_Key  （布弗基） | 此寄存器提供写/读缓冲区R\_KEY寄存器。  宽度 = 8  默认值 = 0 |
| 0x18 + （（i） x 0x0100） | 乌尔曼 | 写入/读取缓冲区长度  （弗尔德布弗伦） | 此寄存器提供写/读缓冲器长度LSB寄存器。  宽度 = 32  默认值 = 0 |
| 0x1C + （（i） x 0x0100） | 乌尔曼 | 访问说明  （访问） | 此寄存器提供 [7：0] 保护域的访问说明和 [31：16] 写入/读取缓冲区长度 MSB 寄存器。默认值为 0。   * [3：0]：访问类型   ° 4'b0000：只读  ° 4'b0001： 只写  ° 4'b0010： 读写  ° 其他值：不支持   * [15：4]：保留 * [31：16]：w\_r\_buf\_len\_msb。寄存器提供写入/读取缓冲区长度 [47：32] |

|  |  |  |  |
| --- | --- | --- | --- |
| **地址** | **访问** | **寄存器名称** | **详** |
| 0x20000 | 乌尔曼 | XRNIC Configuration （XRNICCONF） | 此寄存器为 ERNIC IP 提供基本的全局（非 QP 特定）配置控制   * [0]： ERNIC 启用 * [1]：保留 * [2]：保留 * [4：3]：TX ACK 生成。   ° 00 –（默认）仅在传入数据包中的显式 ACK 请求或超时时生成 ACK  ° 01 – 仅在超时时生成 ACK  ° 10 – 仅在传入数据包中的显式 ACK 请求上生成 ACK   * [5]：启用错误缓冲区。此位设置为启用 错误缓冲区 * [7：6]：保留 * [15：8]：启用的 QP 数。最大序列号 * [31：16]：传出数据包的 UDP 源端口 |
| 0x20004 | 乌尔曼 | XRNIC Advance（XRNIC Advance）  配置  （XRNICADCONF） | 此寄存器为 ERNIC IP 提供高级配置控制。   * [0]：软件覆盖启用。允许对以下内容进行软件写入访问   只读寄存器 – CQHEADn、STATCURRSQPTRn 和  STATRQPIDBn（QP号码在哪里）   * [1]：保留 * [2]： retry\_cnt\_fatal\_dis * [15：3]：保留 * [19：16]：基计数宽度   ° 构成 4096us 的系统时钟的近似数量。  ° 对于 400 MHz 时钟 -->程序十进制 11  ° 对于 200 MHz 时钟 --> 程序十进制 10  ° 对于 125 MHz 时钟 --> 程序十进制 09  ° 对于 100 MHz 时钟 --> 程序十进制 09  ° 对于 N MHz 时钟--->值应为 CLOG2（4.096 \*N）   * [20：23]：保留 * [31：24] 软件替代 QP 编号 |
| 0x20008 | 乌尔曼 | XRNIC\_BUF\_  THRESHOLD\_ROCE | 此寄存器为 RoCE 提供 XON 和 XOFF 配置  交通。   * [15：0]：RoCE 的 XON 值 * [31：16]：RoCE 的 XOFF 值   默认值 = 0x965A |
| 0x2000C | 乌尔曼 | XRNIC\_PAUSE\_CONF | 此寄存器提供 PFC 配置。   * [0]：暂停启用 RoCE * [1]：暂停启用非 RoCE * [7：4]：暂停 RoCE 的优先级 （二进制编码） * [11：8]：暂停非 RoCE 的优先级 （二进制编码） * [12]：禁用优先级检查（适用于TX 数据路径）   默认值 = 0x800 |

|  |  |  |  |
| --- | --- | --- | --- |
| **地址** | **访问** | **寄存器名称** | **详** |
| 0x20010 | 乌尔曼 | MAC XRNIC Address LSB （MACXADDLSB） | 这是本地 （ERNIC） 设备的 MAC 地址，应在 XRNICCONF[0] 设置为 1 之前进行配置。  • [31：0]：MAC 本地地址 LSB |
| 0x20014 | 乌尔曼 | MAC XRNIC Address MSB （MACXADDMSB） | 本地 （ERNIC） 设备的 MAC 地址。应该在 XRNICCONF[0] 设置为 1 之前进行配置。   * [15：0]：MAC 本地地址 MSB * [31：16]：保留 |
| 0x20018 | 乌尔曼 | XRNIC\_BUF\_  THRESHOLD\_NON\_  年 | 此寄存器为非 RoCE 流量提供 XON 和 XOFF 配置。   * [15：0]：非 RoCE 的 XON 值 * [31：16]：非 RoCE 的 XOFF 值   默认值 = 0x965A |
| 0x20020 | 乌尔曼 | IPv6 地址 1  （IPv6XADD1） | 本地 （ERNIC） 设备的 IPv6 地址 [32：0]。应该在 XRNICCONF[0] 设置为 1 之前进行配置。  [31：0]：IP 地址 [31：0] |
| 0x20024 | 乌尔曼 | IPv6 地址 2  （IPv6XADD2） | 本地 （ERNIC） 设备的 IPv6 地址 [63：32]。应在 XRNICCONF[0] 设置为 1 之前进行配置。  [31：0]：IP 地址 [63：32] |
| 0x20028 | 乌尔曼 | IPv6 地址 3  （IPv6XADD3） | 本地 （ERNIC） 设备的 IPv6 地址 [95：64]。应该在 XRNICCONF[0] 设置为 1 之前进行配置。  [31：0]：IP 地址 [95：64] |
| 0x2002C | 乌尔曼 | IPv6 地址 4  （IPv6XADD4） | 本地 （ERNIC） 设备的 IPv6 地址 [127：96]。应该在 XRNICCONF[0] 设置为 1 之前进行配置。  [31：0]：IP 地址 [127：96] |
| 0x20060 | 乌尔曼 | 错误缓冲库  地址（ERRBUFBA） | 此寄存器提供错误缓冲区的基址。ERNIC IP 使用未通过验证的传入数据包更新这些缓冲区。通过向 XRNICCONF[5] 写入 1 来启用对所有验证错误的写入此缓冲区。如果禁用此位d，则只有导致 QP 移动到致命状态的数据包才会写入错误缓冲区。  • [31：0]：错误缓冲区基址 LSB 32 位 |
| 0x20064 | 乌尔曼 | 错误缓冲库  地址 MSB  （ERRBUFBAMSB） | 此寄存器为错误提供 msb 基址。  • [31：0]：错误缓冲区基址 MSB 32 位 |
| 0x20068 | 乌尔曼 | 错误缓冲区大小 （ERRBUFSZ） | 此寄存器提供错误缓冲区的大小   * [15：0]：错误缓冲区数 * [31：16]：每个错误缓冲区的大小 |
| 0x2006C | 反渗透 | 错误缓冲区写入指针 （ERRBUFWPTR） | 当任何错误数据包写入错误缓冲区时，ERNIC IP 会更新此寄存器。指针通知驱动程序收到的错误数据包数。  • [15：0]：为错误缓冲区写入指针门铃 |
| 0x20070 | 乌尔曼 | IPv4 XRNIC Address （IPv4XADD） | 本地 （ERNIC） 设备的 IPv4 地址。  • [31：0]：IPv4 地址 |

|  |  |  |  |
| --- | --- | --- | --- |
| **地址** | **访问** | **寄存器名称** | **详** |
| 0x20078 | 乌尔曼 | 传出 Pkt 错误状态队列基址  （奥普克特尔克巴） | 此寄存器用于配置传出数据包错误状态队列的基址。有关传出数据包错误状态队列的详细信息，请参阅 ERNIC RX 路径。  [31：0]： 传出 PKT 错误状态队列基址 |
| 0x2007C | 乌尔曼 | 传出 Pkt 错误状态队列基址  （OPKTERRQBAMSB） | 此寄存器用于配置传出数据包错误状态队列的 msb 基址。有关传出数据包错误状态队列的详细信息，请参阅 ERNIC RX 路径。  [63：32]： 传出 pkt error 状态队列基址 msb |
| 0x20080 | 乌尔曼 | 传出错误状态  Q 尺寸  （OUTERRSTSQSZ） | [15：0]：传出错误 Q 中的条目数。 |
| 0x20084 | 乌尔曼 | 传出错误状态  Q 写入指针  门铃  （OPTERRSTSQQPTRDB） | 传出错误状态 Q 写入指针门铃。硬件以循环方式不断写入缓冲区。不处理缓冲区溢出。需要在软件中处理。 |
| 0x20088 | 乌尔曼 | 传入 PKT 错误状态队列基址 （IPKTERRQBA） | 此寄存器用于配置传入数据包错误状态队列的基址。有关传入数据包错误状态队列的详细信息，请参阅 ERNIC RX 路径。  [31：0]：传入 PKT 错误状态队列基址 |
| 0x2008C | 乌尔曼 | 传入 pkt 错误状态队列基址 MSB （IPKTERRQBAMSB） | 此寄存器用于为传入的 packet 错误状态队列配置 msb 基址。 |
| 0x20090 | 乌尔曼 | 传入 PKT 错误  状态队列大小  （IPKTERRQSZ） | 此寄存器用于配置传入数据包错误状态队列的大小。   * [15：0]：传入错误 pkt 状态队列条目数 * [31：16]：保留 |
| 0x20094 | 反渗透 | 传入 PKT 错误  状态写入指针  （IPKTERRQWPTR） | 此状态寄存器提供有关 ERNIC IP 接收的传入错误数据包数的信息。  [15：0]：为传入错误状态队列写入指针门铃。ERNIC IP 以循环方式写入队列，而不处理溢出。他需要在SW中处理。 |
| 0x200A0 | 乌尔曼 | 数据缓冲区基础  地址（达特布巴） | ERNIC 使用这些数据缓冲区保存所有传出的 RDMA WRITE 数据，直到远程主机确认该数据。在重新传输的情况下，重试的数据将从这些缓冲区中获取。  [31：0]：数据缓冲区基址 |
| 0x200A4 | 乌尔曼 | 数据缓冲区基础  地址 MSB  （达特布巴姆斯） | ERNIC 使用这些数据缓冲区保存所有传出的 RDMA WRITE 数据，直到远程主机确认该数据。在重新传输的情况下，将从这些缓冲区中提取重试的数据。  [63：32]：数据缓冲区基址 MSB |
| 0x200A8 | 乌尔曼 | 数据缓冲区大小 （DATBUFSZ） | 此寄存器用于配置数据缓冲区的大小。   * [15：0]：数据缓冲区数 * [31：16]：数据缓冲区大小（以字节为单位） |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **地址** | **访问** | **寄存器名称** |  | **详** |
| 0x200AC | 乌尔曼 | 连接 IP 配置 （CON\_IO\_CONF） |  | 此寄存器用于在刚刚启用硬件握手时按门铃。• [15：0]：保留   * [23：16]：队列对 ID * [30：24]：保留 * [31]：连接 IO 就绪指示。此位是只读的，设置为 1'b1 时表示连接 IO 配置已完成并准备好接受另一个 QPID |
| 0x200B0 | 乌尔曼 | 响应错误 pkt 缓冲区基址 （RESPERRPKTBA） |  | ERNIC 使用这些错误缓冲区来保存所有错误响应 pkt msb 基址。重试的地址是从这些缓冲区中提取的。  [31：0]：响应错误 pkt 缓冲区基址 msb |
| 0x200B4 | 乌尔曼 | 响应错误 pkt  缓冲区基址 MSB  （RESPERRPKTBAMSB） |  | ERNIC 使用这些错误缓冲区来保存所有错误响应 pkt msb 基址。重试的地址是从这些缓冲区中提取的。  [31：0]：响应错误 pkt 缓冲区基址 msb |
| 0x200B8 | 乌尔曼 | 响应错误 pkt  缓冲区大小地址 （RESPERRSZ） |  | ERNIC 使用这些错误缓冲区来保存所有错误响应 pkt 大小。它在重试期间使用。[31：0]：响应错误缓冲区大小地址 |
| 0x200BC | 乌尔曼 | 响应错误 pkt 缓冲区大小地址  （RESPERRSZMSB） |  | ERNIC 使用这些错误缓冲区来保存所有错误响应 pkt 大小地址 msb。它在重试期间使用。  [63：32]：响应错误缓冲区大小地址 msb |
|  | | | **全局状态寄存器** | |
| 0x20100 | 反渗透 | 传入发送/读取  响应 Pkt 计数  （INSRRPKTCNT） |  | 这是一个状态寄存器，提供有关传入 RDMA SEND 和 RDMA READ 响应数据包的信息。   * [15：0]：传入发送数据包计数 * [31：16]：传入读取响应数据包计数 |
| 0x20104 | 反渗透 | 传入的确认/疯狂  铂计数  （INAMPKTCNT） |  | 这是一个状态寄存器，提供有关传入确认和管理数据报 （MAD） 数据包的信息。   * [15：0]：传入 ACK 数据包计数 * [31：16]：传入 MAD 数据包计数 |
| 0x20108 | 反渗透 | 传出 IO （SEND/ READ/WRITE） PKT COUNT （OUTIOPKTCNT） |  | 这是一个状态寄存器，提供有关传出 RDMA 发送和 RDMA 读/写数据包的信息。   * [15：0]：传出发送数据包计数 * [31：16]：传出 RDMA 读/写数据包计数 |
| 0x2010C | 反渗透 | 传出的确认/疯狂  铂计数  （OUTAMPKTCNT） |  | 这是一个状态寄存器，提供有关传出确认和 MAD 数据包的信息。   * [15：0]：传出 ACK 数据包计数 * [31：16]：传出 MAD 数据包计数 |

|  |  |  |  |
| --- | --- | --- | --- |
| **地址** | **访问** | **寄存器名称** | **详** |
| 0x20110 | 反渗透 | Last incoming pkt （LSTINPKT） | 此状态寄存器提供上次收到的传入数据包的详细信息。   * [7：0]：最后一个传入数据包的操作码 * [15：8]：最后一个传入数据包的 QPID * [31：16]：最后一个传入数据包的 PSN LSB 位 |
| 0x20114 | 反渗透 | Last Out pkt （LSTOUTPKT） | 此状态寄存器提供最后一个传出数据包的详细信息。   * [7：0]：最后一个传出数据包的操作码 * [15：8]：最后一个传出数据包的 QPID * [31：16]：最后一个传出数据包的 PSN LSB 位 |
| 0x20118 | 反渗透 | 传入无效/  重复的 pkt 计数  （ININVDUPCNT） | 此状态寄存器提供有关传入无效或重复数据包的信息。   * [15：0]：传入无效数据包计数 * [31：16]：传入重复数据包计数 |
| 0x2011C | 反渗透 | 传入的 NAK pkt  Status （INNCKPKTSTS） | 此状态寄存器提供传入响应程序未就绪 （RNR） NAK、重试计数已过期或收到的其他 NAK 的详细信息。   * [7：0]：传入 NAK 数据包的 QPID * [15：8]：保留 * [31：16]：传入 NAK 数据包计数 |
| 0x20120 | 反渗透 | 传出 RNR pkt  地位  （OUTRNRPKTSTS） | 此状态寄存器提供已发送的传出响应方未就绪 （RNR） NAK 的详细信息。   * [7：0]：传出 RNR 数据包的 QPID * [15：8]：保留 * [31：16]：发送的 RNR 数据包计数 |
| 0x20124 | 反渗透 | WQE 处理器状态 （WQEPROCSTS） | 此状态寄存器提供 WQE 处理器引擎的状态，并用于调试目的。   * [2：0]：WQE 处理器状态 * [3]：保留 * [4]：重试缓冲区不可用 * [5]：内部先进先出空状态 * [6]：缓冲区管理器忙音 * [11：7]：保留 * [12]：来自CMAC的背压指示。粘性钻头 * [15：12]：标头缓冲区管理器 FSM 状态 * [23：16]：缓冲区尾部指针 * [31：24]：缓冲区头指针 |
| 0x2012C | 反渗透 | QP 管理器状态 （QPMSTS） | 此状态寄存器提供 QP 管理器的状态，并用于调试目的。• [0]：WQE 缓存已满   * [1]：WQE 缓存为空 * [15：2]：保留 * [31：16]：处理的 WQE 计数 |

|  |  |  |  |
| --- | --- | --- | --- |
| **地址** | **访问** | **寄存器名称** | **详** |
| 0x20130 | 反渗透 | 传入全部/丢弃的 pkt 计数  （INALLDRPPKTCNT） | 此状态寄存器提供所有传入和丢弃的数据包的详细信息。   * [15：0]：所有传入数据包计数 * [31：16]：传入丢弃的数据包计数 |
| 0x20134 | 反渗透 | Incoming NAK pkt count （INNAKPKTCNT） | 此状态寄存器提供所有传入 NAK 数据包的详细信息。   * [15：0]：传入 NAK 数据包计数 * [31：16]：保留 |
| 0x20138 | 反渗透 | 传出 NAK pkt 计数  （OUTNAKPKTCNT） | 此状态寄存器提供所有传出 NAK 数据包的详细信息。   * [15：0]：传出 NAK 数据包计数 * [31：16]：保留 |
| 0x2013C | 反渗透 | 响应处理程序  状态（RESPHNDSTS） | 此状态寄存器提供响应处理程序模块的状态，并用于调试目的。   * [7：0]：仲裁 QP 指数 * [13：8]：响应处理程序 FSM 状态 * [16：14]：保留 * [17]：要处理的确认 * [31：18]：保留 |
| 0x20140 | 反渗透 | 重试计数状态 （RETRYCNTSTS） | 此状态寄存器提供重新传输的数据包的详细信息。  [31：0] 重试计数 |
| 0x20174 | 反渗透 | 传入 CNP 数据包计数 （INCNPPKTCNT） | 传入 CNP 数据包计数 |
| 0x20178 | 反渗透 | 传出 CNP 数据包  计数  （对外） | 传出 CNP 数据包计数 |
| 0x2017C | 反渗透 | 传出读取响应数据包计数  （OUTRDRSPPKTCNT） | 传出读取响应数据包计数 |
| 0x20180 | 乌尔曼 | 中断使能 （INTEN） | 该寄存器为 ERNIC IP 生成的中断提供配置控制。   * [0]：传入数据包验证错误中断启用 * [1]：传入 MAD 数据包接收中断启用 * [2]：保留 * [3]：RNR NACK 生成的中断使能 * [4]： WQE 完成中断启用（对于设置了 QPCONF[3] 位的 QP 置位） * [5]：在 SEND 队列中断中断中发布的非法操作码启用 * [6]：RQ 数据包接收中断启用（对于设置了 QPCONF[2] 位的 QP 断言 * [7]：致命错误接收中断启用 * [8]：CNP 调度的中断启用 |

|  |  |  |  |
| --- | --- | --- | --- |
| **地址** | **访问** | **寄存器名称** | **详** |
| 0x20184 | W1C | 中断状态 （INTSTS） | 此状态寄存器提供断言中断的原因。   * [0]：传入数据包验证错误中断 * [1]：传入的 MAD 数据包收到中断 * [2]：保留 * [3]：RNR NACK 生成的中断 * [4]：WQE 完成中断（对于设置了 QPCONF[3] 位的 QP 断言） * [5]：在 SEND 队列中断中发布非法操作码 * [6]：RQ 数据包接收中断（对于设置了 QPCONF[2] 位的 QP 断言） * [7]：收到致命错误中断 * [8]： CNP 调度的中断启用 |
| 0x20190 | WC1 | Rcv Q 中断状态  （按位）QP 0-31  （RQINTSTS1） | 此寄存器提供 QP 0 到 31 的 RQ 中断状态。位 [i] 提供 RQi 的中断状态，其中 i=0 到 31。  [i]：RQ i 的中断状态 |
| 0x20194 | WC1 | Rcv Q 中断状态  （按位）QP 32-63  （RQINTSTS2） | 该寄存器提供 QP 32 至 63 的 RQ 中断状态。位 [i] 提供 RQ（i+32） 的中断状态，其中 i=0 到 31。  [i]：RQ（i+32） 的中断状态 |
| 0x20198 | WC1 | Rcv Q 中断状态  （按位）QP 64-95  （RQINTSTS3） | 该寄存器提供 QP 64 至 95 的 RQ 中断状态。位 [i] 提供 RQ（i+64） 的中断状态，其中 i=0 到 31。  [i]：RQ（i+64） 的中断状态 |
| 0x2019C | WC1 | Rcv Q 中断状态  （按位）QP 96-127  （RQINTSTS4） | 该寄存器提供 QP 96 至 127 的 RQ 中断状态。位 [i] 提供 RQ（i+96） 的中断状态，其中 i=0 到 31。  [i]：RQ（i+96） 的中断状态 |
| 0x201A0 | WC1 | Rcv Q 中断状态  （按位）QP 128-159  （RQINTSTS5） | 该寄存器提供 QP 128 至 159 的 RQ 中断状态。位 [i] 提供 RQ（i+128） 的中断状态，其中 i=0 到 31。  [i]：RQ（i+128） 的中断状态 |
| 0x201A4 | WC1 | Rcv Q 中断状态  （按位）QP 160-191  （RQINTSTS6） | 该寄存器提供 QP 160 至 191 的 RQ 中断状态。位 [i] 提供 RQ（i+160） 的中断状态，其中 i=0 到 31。  [i]：RQ（i+160） 的中断状态 |
| 0x201A8 | WC1 | Rcv Q 中断状态  （按位）QP 192-223  （RQINTSTS7） | 该寄存器提供 QP 192 至 223 的 RQ 中断状态。位 [i] 提供 RQ（i+192） 的中断状态，其中 i=0 到 31。  [i]：RQ（i+192） 的中断状态 |
| 0x201AC | WC1 | Rcv Q 中断状态  （按位）QP 224-255  （RQINTSTS8） | 该寄存器提供 QP 224 至 255 的 RQ 中断状态。位 [i] 提供 RQ（i+224） 的中断状态，其中 i=0 到 31。  [i]：RQ（i+224） 的中断状态 |
| 0x201B0 | WC1 | 完成 Q 中断状态（按位） QP 0-31  （CQINTSTS1） | 该寄存器提供 QP 0 到 31 的 CQ 中断状态。位 [i] 提供 CQi 的中断状态，其中 i=0 到 31。  [i]：CQ i 的中断状态 |
| 0x201B4 | WC1 | 完成 Q 中断状态（按位） QP 32-63  （CQINTSTS2） | 该寄存器提供 QP 32 至 63 的 CQ 中断状态。位 [i] 提供 CQ（i+32） 的中断状态，其中 i=0 到 31。  [i]：CQ（i+32） 的中断状态 |

|  |  |  |  |
| --- | --- | --- | --- |
| **地址** | **访问** | **寄存器名称** | **详** |
| 0x201B8 | WC1 | 完成 Q 中断状态（按位） QP 64-95  （CQINTSTS3） | 该寄存器提供 QP 64 至 95 的 CQ 中断状态。位 [i] 提供 CQ（i+64） 的中断状态，其中 i=0 到 31。  [i]：CQ（i+64） 的中断状态 |
| 0x201BC | WC1 | 完成 Q 中断状态（按位） QP 96-127  （CQINTSTS4） | 该寄存器提供 QP 96 至 127 的 CQ 中断状态。位 [i] 提供 CQ（i+96） 的中断状态，其中 i=0 到 31。  [i]：CQ（i+96） 的中断状态 |
| 0x201C0 | WC1 | 完成 Q 中断状态（按位） QP 128-159  （CQINTSTS5） | 该寄存器提供 QP 128 至 159 的 CQ 中断状态。位 [i] 提供 CQ（i+128） 的中断状态，其中 i=0 到 31。  [i]：CQ（i+128） 的中断状态 |
| 0x201C4 | WC1 | 完成 Q 中断状态（按位） QP 160-191  （CQINTSTS6） | 此注册表提供 QP 160 到 191 的 CQ 中断状态。位 [i] 提供 CQ（i+160） 的中断状态，其中 i=0 到 31。  [i]：CQ（i+160） 的中断状态 |
| 0x201C8 | WC1 | 完成 Q 中断状态（按位） QP 192-223  （CQINTSTS7） | 该寄存器提供 QP 192 至 223 的 CQ 中断状态。位 [i] 提供 CQ（i+192） 的中断状态，其中 i=0 到 31。  [i]：CQ（i+192） 的中断状态 |
| 0x201CC | WC1 | 完成 Q 中断状态（按位） QP 224-255  （CQINTSTS8） | 该寄存器提供 QP 224 至 255 的 CQ 中断状态。位 [i] 提供 CQ（i+224） 的中断状态，其中 i=0 到 31。  [i]：CQ（i+224） 的中断状态 |
| 0x201D0 | 反渗透 | CNP 打包的计划中断状态（按位） QP 0-31  （CNPSCHDSTS1REG） | 此寄存器为 QP 0 到 31 提供 CNP 计划中断。 |
| 0x201D4 | 反渗透 | CNP 打包计划中断状态（按位） QP 32-63  （CNPSCHDSTS2REG） | 此寄存器为 QP 32 至 64 提供 CNP 计划中断。 |
| 0x201D8 | 反渗透 | CNP 打包的计划中断状态（按位） QP 64-95  （CNPSCHDSTS3REG） | 此寄存器为 QP 64 至 95 提供 CNP 计划中断。 |
| 0x201DC | 反渗透 | CNP 打包的计划中断状态（按位） QP 96-127  （CNPSCHDSTS4REG） | 此寄存器为 QP 96 至 127 提供 CNP 计划中断。 |
| 0x201E0 | 反渗透 | CNP 打包的计划中断状态（按位） QP 128-159  （CNPSCHDSTS5REG） | 此寄存器为QP 128 到 159 提供 CNP 计划中断。 |

|  |  |  |  |
| --- | --- | --- | --- |
| **地址** | **访问** | **寄存器名称** | **详** |
| 0x201E4 | 反渗透 | CNP 打包的计划中断状态（按位） QP 160-191  （CNPSCHDSTS6REG） | 该寄存器为 QP 160 至 191 提供 CNP 计划中断。 |
| 0x201E8 | 反渗透 | CNP 打包的计划中断状态（按位） QP 192-223  （CNPSCHDSTS7REG） | 此寄存器为 QP 192 到 223 提供 CNP 计划中断。 |
| 0x201EC | 反渗透 | CNP 打包的计划中断状态（按位） QP 224-255  （CNPSCHDSTS8REG） | 该寄存器为 QP 224 至 255 提供 CNP 计划中断。 |

|  |  |  |  |
| --- | --- | --- | --- |
| **地址** | **访问** | **寄存器名称** | **详** |
| **每个 QP 寄存器** | | | |
| 0x20200 +  （（I-1） x  0x0100） | 乌尔曼 | QP 配置 QPi （QPCONFi） | 该寄存器定义了 QP 的基本配置。此寄存器基于参数 C\_NUM\_QP 生成（其中 i = 1 到 C\_NUM\_QP）。每个 QP 的寄存器偏移量为 0x20200，后续偏移的0x20204以此类推。   * [0]：QP 启用 – 应为所有活动 QP 设置 t o 1。禁用的 QP 将无法接收或传输数据包。 * [2]：RQ 中断启用 – 启用后，允许为接收队列上收到的每个新数据包生成接收队列中断 * [3]：CQ 中断启用 – 启用后，允许为每个发送工作队列条目完成生成完成队列中断 * [4]：硬件握手禁用 – 重置为 0 时，此位启用硬件握手端口以进行门铃交换。如果设置，则所有门铃值都将通过 AXI4 或 AXI4-Lite 接口写入。 * [5]：CQE 写启用 – 设置此位后，启用完成队列条目写入。重置此位时，将禁用写入。可以启用 CQE 写入以调试失败的完成。 * [6]：QP under 恢复。这个位需要在致命的清除过程中设置。 * [7]：为 IPv4 或 IPv6 配置的 QP   ° 0 - IPv4  ° 1 - IPv6   * [10：8]：路径 MTU   ° 000 – 256B （默认）  ° 001 – 512B  ° 010 – 1024B  ° 011 – 2048B  ° 100 - 4096B  ° 101 到 111 - 保留   * [31：16]：RQ 缓冲区大小 （256B 的倍数） 。这是请求中每个缓冲区元素的大小，而不是整个请求的大小。   对于预期行为，编程值应为 2 的幂。 |
| 0x20204 +  （（I-1） x  0x0100） | 乌尔曼 | QP 高级配置 QPi  （QPADVCONFi） | 此寄存器基于参数 C\_NUM\_QP 生成（其中 i = 2 到 C\_NUM\_QP）。QP1 不存在此寄存器。   * [5：0]：流量类 （保留默认重置值） * [15：8]： 是时候了 * [31：16]：分区键 |
| 0x20208 +  （（I-1） x  0x0100） | 乌尔曼 | Rcv Q 缓冲区基址 QPi （RQBAi） | 此寄存器基于参数 C\_NUM\_QP 生成（其中 i = 1 到 C\_NUM\_QP）。此寄存器提供 RDMA 接收队列缓冲区的基址。  [31：8]：接收 Q 缓冲区基址地址 （256 B 对齐） 。256 B 是总计（单个 rq buffer 元素大小）\* rq\_depth |

|  |  |  |  |
| --- | --- | --- | --- |
| **地址** | **访问** | **寄存器名称** | **详** |
| 0x202C0 +  （I-1） x  0x0100） | 乌尔曼 | Rcv Q 缓冲基座  地址 QPi （RQBAMSBi） | 此寄存器基于参数 C\_NUM\_QP 生成（其中 i = 1 到 C\_NUM\_QP）。此寄存器提供 RDMA 接收队列缓冲区的 msb 基址。  [63：32]：接收 Q 缓冲区基址地址 （256 B 对齐） 。256 B 是总值（单个 rq 缓冲区元素大小）\* rq\_depth |
| 0x20210 +  （（I-1） x  0x0100） | 乌尔曼 | SEND Q 基址 QPi （SQBAi） | 此寄存器基于参数 C\_NUM\_QP 生成（其中 i = 1 到 C\_NUM\_QP）。此寄存器提供 RDMA 发送队列缓冲区的基址。[31：5]：发送 Q 基址 （32 B 对齐） |
| 0x202C8 +  （（I-1） x  0x0100） |  | SEND Q 基址 msb QPi （SQBAMSBi） | 此寄存器基于参数 C\_NUM\_QP 生成（其中 i = 1 到 C\_NUM\_QP）。此寄存器提供 RDMA 发送队列缓冲区的 msb 基址。[63：32]：发送 Q 基址 （32 B 对齐） |
| 0x20218 +  （（I-1） x  0x0100） | 乌尔曼 | CQ 基址 QPi （CQBAi） | 此寄存器基于参数 C\_NUM\_QP 生成（其中 i = 1 到 C\_NUM\_QP）。此寄存器提供 RDMA 完成队列缓冲区的基址。  [31：5]：发送 CQ 基址 （32 B 对齐） |
| 0x202D0 +  （（I-1） x  0x0100） | 乌尔曼 | CQ 基址 msb QPi （CQBAMSBi） | 此寄存器基于参数 C\_NUM\_QP 生成（其中 i = 1 到 C\_NUM\_QP）。此寄存器提供 RDMA 完成队列缓冲区的 msb 基址。[63：32]：发送 CQ 基址 （32 B 对齐） |
| 0x20220 +  （（I-1） x  0x0100） | 乌尔曼 | Rcv Q 写入 指针 DB 地址 QPi  （RQWPTRDBADDi） | 此寄存器基于参数 C\_NUM\_QP 生成（其中 i = 1 到 C\_NUM\_QP）。此寄存器提供接收队列门铃寄存器的地址。收到新的传入 RDMA SEND 数据包后，ERNIC IP 会更新此寄存器指向的地址中的 RQ 门铃值。  [31：0]：RCV Q 写入指针门铃地址 |
| 0x20224 +  （（I-1） x  0x0100） | 乌尔曼 | Rcv Q 写入指针 DB 地址 msb QPi  （RQWPTRDBADDMSBi） | 此寄存器基于参数 C\_NUM\_QP 生成（其中 i = 1 到 C\_NUM\_QP）。此寄存器提供接收队列门铃寄存器的 msb 地址。收到新的传入 RDMA SEND 数据包后，ERNIC IP 会更新此寄存器指向的地址中的 RQ 门铃值。  [63：32]：Rcv Q 写入指针门铃地址 msb |
| 0x20228 +  （（I-1） x  0x0100） | 乌尔曼 | CQ DB 地址 QPi （CQDBADDi） | 此寄存器基于参数 C\_NUM\_QP 生成（其中 i = 1 到 C\_NUM\_QP）。此注册表ster 提供完成队列门铃寄存器的地址。完成新的 SEND 工作队列条目后，ERNIC IP 将更新此寄存器指向的地址中的 CQ 门铃值。  [31：0]：发送CQ门铃地址 |
| 0x2022C +  （（I-1） x  0x0100） | 乌尔曼 | CQ DB 地址 QPi （CQDBADDMSBi） | 此寄存器基于参数 C\_NUM\_QP 生成（其中 i = 1 到 C\_NUM\_QP）。此寄存器提供完成队列门铃寄存器的 MSB 地址。完成新的 SEND 工作队列条目后，ERNIC IP 将更新此寄存器指向的地址中的 CQ 门铃值。  [63：32]：发送CQ门铃地址 |

|  |  |  |  |
| --- | --- | --- | --- |
| **地址** | **访问** | **寄存器名称** | **详** |
| 0x20230 +  （（I-1） x  0x0100） | 反渗透 | CQ 头指针 QPi （CQHEADi） | 此寄存器基于参数 C\_NUM\_QP 生成（其中 i = 1 到 C\_NUM\_QP）。此状态寄存器提供发送完成队列门铃值，并提供有关已完成的发送 WQE 的信息。这是由 ERNIC IP 写入 CQDBADDi 寄存器指向的地址的门铃值。   * [15：0]：CQ 头指针 * [31：0]：保留 |
| 0x20234 +  （（I-1） x  0x0100） | 乌尔曼 | RQ Consumer Index QPi （RQCIi） | 此寄存器基于参数 C\_NUM\_QP 生成（其中 i = 1 到 C\_NUM\_QP）。此寄存器由目标应用程序在使用新的接收队列条目时更新。一旦使用，RQ 条目可以被 ERNIC IP 覆盖。RQCI指数 也可以通过边带接口更新。   * [15：0]：RCV Q 消费者指数门铃 * [31：16]：保留 |
| 0x20238 +  （（I-1） x  0x0100） | 乌尔曼 | SQ 生产者指数 QPi （SQPIi） | 此寄存器基于参数 C\_NUM\_QP 生成（其中 i = 1 到 C\_NUM\_QP）。目标应用程序在发布新的发送工作队列条目时会更新此寄存器。SQPI 也可以通过边带接口进行更新。发送队列 是一个循环缓冲区。相应 QP 的 CQHEAD 寄存器提供有关已完成并且可以由目标应用程序覆盖的工作队列条目的信息。   * [15：0]：发送 Q 生产者索引门铃 * [31：16]：保留 |
| 0x2023C +  （（I-1） x  0x0100） | 乌尔曼 | Q Depth QPi （QDEPTHi） | 此寄存器基于参数 C\_NUM\_QP 生成（其中 i = 1 到 C\_NUM\_QP）。此寄存器定义发送、完成和接收队列的队列深度。   * [15：0]：发送 Q 深度 （发送相同深度的 CQ 将） * [31：16]：接收 Q 深度 |
| 0x20240 +  （（I-1） x  0x0100） | 乌尔曼 | SEND Q PSN for QPi （SQPSNi） | 此寄存器在连接时由软件初始化。之后，硬件会为每个传出数据包更新它，并且不应由软件更新。此寄存器基于参数 C\_NUM\_QP 生成（其中 i = 2 到 C\_NUM\_QP）。QP1 不存在此寄存器。  [23：0]：发送 Q PSN |
| 0x20244 +  （（I-1） x  0x0100） | 乌尔曼 | Last RQ req for QPi （LSTRQREQi） | 此寄存器提供最后传入的 RQ 数据包详细信息。此寄存器基于参数 C\_NUM\_QP 生成（其中 i = 2 到 C\_NUM\_QP）。QP1 不存在此寄存器。   * [23：0]：RCV Q PSN * [31：24]：RCV Q 操作码 |
| 0x20248 +  （（I-1） x  0x0100） | 乌尔曼 | QPi 的目标 QP 配置  （DESTQPCONFi） | 此寄存器基于参数 C\_NUM\_QP 生成（其中 i = 1 到 C\_NUM\_QP）。此寄存器在连接时由软件配置，并提供连接到此 QP 的远程 QPID。来自此 QP 的所有传出数据包都与此 QPID 一起作为目标 QPID 发送。  [23：0]：目标连接的 QPID |

|  |  |  |  |
| --- | --- | --- | --- |
| **地址** | **访问** | **寄存器名称** | **详** |
| 0x20250 +  （（I-1） x  0x0100） | 乌尔曼 | MAC 目标地址 LSB QPi （MACDESADDLSBi） | 此寄存器基于参数 C\_NUM\_QP 生成（其中 i = 1 到 C\_NUM\_QP）。此寄存器在连接时由 SW 配置，并提供连接到此 QP 的远程主机的 MAC 地址。来自此 QP 的所有传出数据包都使用此 MAC 地址 （LSB） 作为目标 MAC 地址发送。  [31：0]：MAC 目标地址 LSB |
| 0x20254 +  （（I-1） x  0x0100） | 乌尔曼 | MAC 目标地址 MSB QPi （MACDESADDMSBi） | 此寄存器基于参数 C\_NUM\_QP 生成（其中 i = 1 到 C\_NUM\_QP）。此寄存器在连接时由 SW 配置，并提供连接到此 QP 的远程主机的 MAC 地址。来自此 QP 的所有传出数据包都使用此 MAC 地址 （MSB） 作为目标 MAC 地址发送。  [15：0]：MAC 目标地址 MSB |
| 0x20260 +  （（I-1） x  0x0100） | 乌尔曼 | IP 目标地址 1 QPi （IPDESADDR1i） | 此寄存器基于参数 C\_NUM\_QP 生成（其中 i = 1 到 C\_NUM\_QP）。如果此 QP 的远程主机是使用 IPv4 协议配置的，则此寄存器指示 IPv4 地址。如果远程主机是使用 IPv6 协议 ol 配置的，则此寄存器会指示 IPv6 目标地址。  [31：0]：IP 目标地址 LSB1 |
| 0x20264 +  （（I-1） x  0x0100） | 乌尔曼 | IP 目标地址 2 QPi （IPDESADDR2i） | 此寄存器基于参数 C\_NUM\_QP 生成（其中 i = 1 到 C\_NUM\_QP）。如果使用 IPv6 协议配置远程主机，则此寄存器指示 IPv6 目标地址 [63：32]。如果此 QP 的远程主机是使用 IPv4 协议配置的，则不使用此寄存器。  [31：0]：IP 目标地址 LSB2 |
| 0x20268 +  （（I-1） x  0x0100） | 乌尔曼 | IP 目标地址 3 QPi （IPDESADDR3i） | 此寄存器基于参数 C\_NUM\_QP 生成（其中 i = 1 到 C\_NUM\_QP）。如果使用 IPv6 协议配置远程主机，则此寄存器指示 IPv6 目标地址 [95：64]。如果此 QP 的远程主机是使用IPv4 协议配置的，则不使用此寄存器。  [31：0]：IP 目标地址 MSB1 |
| 0x2026C +  （（I-1） x  0x0100） | 乌尔曼 | IP 目标地址 4 QPi （IPDESADDR4i） | 此寄存器基于参数 C\_NUM\_QP 生成（其中 i = 1 到 C\_NUM\_QP）。如果使用 IPv6 协议配置远程主机，则此寄存器指示 IPv6 目标地址 [127：96]。如果此 QP 的远程主机是使用 IPv4 协议配置的，则不使用此寄存器。  [31：0]：IP 目标地址 MSB2 |

|  |  |  |  |
| --- | --- | --- | --- |
| **地址** | **访问** | **寄存器名称** | **详** |
| 0x2024C +  （（I-1） x  0x0100） | 乌尔曼 | 超时配置  注册 QPi  （超时会议i） | 此寄存器基于参数 C\_NUM\_QP 生成（其中 i = 1 到 C\_NUM\_QP）。它提供超时配置 QPi。QP1 不存在此寄存器。   * [5：0]：超时值 * [7：6]：保留 * [10：8]：最大重试计数 * [13：11]：最大 RNR 重试计数 * [15：14]：保留 * [20：16]：传出数据包的 RNR NACK 超时值 * [31：21]：保留 |
| 0x20280 +  （（I-1） x  0x0100） | 反渗透 | 状态发件人序列号 QPi （STATSSNi） | 此寄存器基于参数 C\_NUM\_QP 生成（其中 i = 2 到 C\_NUM\_QP）。QP1 不存在此寄存器。  [23：0]：当前传出 SSN （与传出 MSN 相同） |
| 0x20284 +  （（I-1） x  0x0100） | 反渗透 | 状态消息序列号 QPi （STATMSN） | 此寄存器基于参数 C\_NUM\_QP 生成（其中 i = 2 到 C\_NUM\_QP）。QP1 不存在此寄存器。  [23：0]：当前预期的传入 MSN |
| 0x20288 +  （（I-1） x  0x0100） | 反渗透 | Status QPi （STATQPi） | 此寄存器提供每个 QP 的 QP 状态。此寄存器基于参数 C\_NUM\_QP 生成（其中 i = 1 到 C\_NUM\_QP）。   * [0]：QP 处于致命状态 * [1]： Rcv Q ovfl （传出 RNR NACK） * [2]：发送 Q 满 * [3]：出色的 Q 满 * [4]： CQ 先进先出已满 * [8：5]：保留 * [9]：将 Q 发送为空。此位表示没有要处理的 SQE。但是，这并不意味着远程主机已确认所有 SEND WQE。 * [10]：未完成的 Q 空 * [11]： QP 数据包已重试 * [15：12]：保留 * [22：16]：收到 NACK 综合征 （读/写） * [23]：保留 * [26：24]：当前重试计数 （读/写） * [27]：保留 * [30：28]：inc resp 数据包的当前 RNR 纳克计数 （读/写） * [31]：保留 |
| 02x028C +  （（I-1） x  0x0100） | 反渗透 | 状态 当前 SQ PTR 正在处理 （STATCURSQPTRi） | 此寄存器基于参数 C\_NUM\_QP 生成（其中 i = 2 到 C\_NUM\_QP）。  [15：0]：当前正在处理的 SQ 指针。这显示了 a重新未完成并等待远程主机响应的 WQE 数。 |

|  |  |  |  |
| --- | --- | --- | --- |
| **地址** | **访问** | **寄存器名称** | **详** |
| 0x20290 +  （（I-1） x  0x0100） | 反渗透 | 状态响应 PSN for QPi （STATRESPSNi） | 此寄存器基于参数 C\_NUM\_QP 生成（其中 i = 2 到 C\_NUM\_QP）。QP1 不存在此寄存器。  [23：0]：预期响应 PSN |
| 0x20294 +  （（I-1） x  0x0100） | 反渗透 | QPi 的状态 RQ 缓冲区当前地址 （STATRQBUFCAi） | 此寄存器基于参数 C\_NUM\_QP 生成（其中 i = 2 到 C\_NUM\_QP）。  [31：8]：接收 Q 缓冲电流地址 （256 B 对齐） |
| 0x202D8 +  （（I-1） x  0x0100） | 反渗透 | QPi 的状态 RQ 缓冲区当前 MSB 地址  （STATRQBUFCAMSBi） | 此寄存器基于参数 C\_NUM\_QP 生成（其中 i = 2 到 C\_NUM\_QP）。  [63：32]：接收 Q 缓冲电流地址 msb （256 B 对齐） |
| 0x20298 +  （（I-1） x  0x0100） | 反渗透 | 发布到 QPi （STATWQEi） 的 WQE 状态 | 此寄存器基于参数 C\_NUM\_QP 生成（其中 i = 2 到 C\_NUM\_QP）。  [15：0]：QP MGR 为此 QP 推送的 WQE 计数 |
| 0x2029C +  （（I-1） x  0x0100） | 反渗透 | Status RQ Producer index DB QPi （STATRQPIDBi） | 此寄存器基于参数 C\_NUM\_QP 生成（其中 i = 2 到 C\_NUM\_QP）。  [15：0]：RCV Q 生产者指数数据库 |
| 0x202B0 +  （（I-1） x  0x0100） | 乌尔曼 | 保护域号 | 此寄存器基于参数 C\_NUM\_QP 生成（其中 i = 2 到 C\_NUM\_QP）。该寄存器是 24 位的，包含分配给 QP 的 PD 编号。  ***注意：*** 每个 QP 只能关联一个内存区域。 |

# *第 3 章* 核心设计

本章包括指南和其他信息，以促进使用 ERNIC IP 核进行设计。

## 一般设计准则

典型的 ERNIC 子系统将包括一个或多个 MAC 硬/软 IP。还需要 AXI 互连来连接 ERNIC IP 公开的各种 AXI 接口。DRAM 可以是解决方案的一部分，这也需要 DRAM 控制器进行实例化。

X-参考目标 - 图 3-1

复员方案

ERNIC 目标 IP

pkt\_hndler I/F

wQe\_proc l/f

QP 管理公司 l/f

Resp Hndler L/F

wqe\_proc\_wr\_retry

M

M

M

M

M

M

M

S

M

S

S

M

non\_roce\_dma\_m\_axis\_I/楼

中国驻地协调员

S

s\_axi\_lite

客户逻辑

pkt\_hndler rresp\_I/楼

RQPI 边带

RQCI 边带

SQCI边带

平方带边带

AXI4-流

AXI4

AXI4-精简版

客户侧带

X19883-010621

AXI

二甲基亚胺

non\_roce\_dma\_s\_axis\_I/楼

non\_roce\_cmac\_s\_axis\_I/楼

roce\_cmac\_s\_axis\_I/楼

cmac\_m\_axis\_I/楼

S

S

M

*图 3-1：* **ERNIC 接口**

图 3-1 显示了各种 ERNIC IP 接口。显示为与 DDR 接口的接口可以与任何内存映射区域接口。有关每个接口的详细信息，请参阅表 2-5。AXI4 和 AXI4 流接口的宽度为 512 位，主要用于数据传输。ERNIC IP 提供边带接口，允许第  *3 章：***使用内核进行设计**

高效交换队列对相关门铃。可以根据配置为每个队列对 （QP） 启用或禁用这些边带接口。

ERNIC IP 有一个 AXI4-Lite 从接口用于访问寄存器空间。 下表显示了此从接口所需的内存映射的详细信息。

*表 3-1：* **从接口的地址空间分配要求**

|  |  |  |  |
| --- | --- | --- | --- |
| **从接口** | **大小** | **单位** | **描述** |
| ERNIC AXI4-Lite 从机接口 | 256 | KB | ERNIC 寄存器接口 |

除此之外，IP还需要为某些特定的数据结构分配一些内存区域。这些内存区域可以映射到本地DRAM或AXI BRAM或任何其他内存映射从站。根据ERNIC 应达到的线速，确保 se 内存接口上有足够的带宽。

*表 3-2：* **ERNIC IP 内存要求**

|  |  |  |  |
| --- | --- | --- | --- |
| **内存区域** | **大小** | **单位** | **描述** |
| 错误缓冲区 | 64 | KB | 256 个数据包，每个数据包 256 字节。未通过数据包验证的数据包将与 4 字节错误综合症一起发送到错误缓冲区 |
| 发送队列 | 2 | 兆字节 | 256 个 QP，深度为 128 个位置，每个 SQE 为 64 字节 |
| 接收队列 | 32 | 兆字节 | 256 个 QP，深度 128 个位置，每个 RQE 是  每个 1024 字节 |
| 发送完成队列 | 128 | KB | 256 个 QP，深度为 128 个位置，每个 CQE 为 4 个字节 |
| 写入重试缓冲区 | 16 | 兆字节 | 每个 4K 大小的缓冲区，用于 256 个 QP，每个 QP 最多包含 16 个未完成事务 |

**笔记：**

1. 使用 256 个 QP 的 ERNIC 时，请使用 Vivado 实施策略 — Performance\_RefinePlacement。

## 中断

ERNIC IP 提供一条中断线，该线在 INTEN 寄存器中定义的不同中断状态信号上生成。这些中断线中的每一个都可以通过向中断启用 INTEN 寄存器的相应位写入 1 来使能。收到 ORed中断时，软件可以读取INTSTS寄存器以了解中断的原因。

中断位 4 和 6 分别通知驱动程序有关 WQE 完成或传入 RDMA SEND 的信息。这些中断可以在QP 的基础上启用或禁用。QPCONFi 寄存器的位 [2] 允许选择性地启用每个 QP 的接收队列中断。

同样，QPCONFi 寄存器的位 [3] 允许选择性地启用发送完成队列中断。通常，需要软件处理的 QP 应启用此选项。

由硬件直接处理的 QP 将通过硬件通知 *第 3 章：* **使用核心进行设计**

可以禁用握手端口和相应的中断使能位。此类 QP 应将h ardware 握手禁用 QPCONFi[4] 位重置为 0。

RQINTSTSn 和 CQINTSTSn 寄存器提供有关具有待处理的 RQ 或 CQ 条目的 QP 的按位信息。

软件应分别在接收 RQ 或 CQ 接口时读取这些寄存器，以了解需要维修的 QP。成功处理后，应通过向相应的位写入 1 来清除这些中断状态。

## 时钟

两个时钟暴露在 ERNIC IP 的顶部。它们是：AXI4 时钟 和 AXI4-Lite 时钟。所有寄存器访问都在 AXI4-Lite 时钟上工作，而其余逻辑则在

AXI4 时钟。通常，AXI4 时钟的频率较高（高达 200 MHz），而 AXI4-Lite 接口的时钟频率可能较低（AXI4 分 频时钟，分频时钟边沿与 AXI4 时钟对齐）。但是，这些时钟在 ERNIC IP 内被视为同步时钟，并且应从同一时钟源生成。

## 重 置

ERNIC IP 需要两次/电平有效复位，分别同步到两个时钟域。

# *第4章* 设计流程步骤

本章介绍自定义和生成内核、约束内核以及特定于此 IP 内核的仿真、综合和输入步骤。有关标准 Vivado 设计流程和 IP 集成器的更多详细信息，请参阅以下 Vivado® 设计套件用户指南：

* *Vivado 设计套件用户指南：使用 IP 积分器设计 IP* 子系统 （UG994） [参考文献 2]
* *Vivado 设计套件用户指南： 利用 IP 进行设计* （UG896） [参考文献 3]
* *Vivado 设计套件用户指南：入门* （UG910） [参考文献 4]
* *Vivado 设计套件用户指南： 逻辑仿真* （UG900） [参考文献 5]

## 自定义和生成核心

本节包含有关使用 Xilinx 工具在 Vivado 设计套件中定制和生成内核的信息。

如果要在 Vivado IP 集成器中定制和生成内核，请参阅 *Vivado 设计套件用户指南：使用 IP 集成器设计 IP 子系统* （UG994） [参考文献 2] 了解详细信息。IP 集成商可能会在验证或生成设计时自动计算某些配置值。要检查值是否更改，请参阅本章中参数的说明。要查看参数值，请在 Tcl 控制台中运行 validate\_bd\_design 命令。

您可以使用 以下步骤为与 IP 核关联的各种参数指定值，从而自定义设计中使用的 IP：

1. 从 Vivado IP 目录中选择 IP。
2. 双击选定的 IP 或从工具栏或右键k 菜单中选择“**自定义 IP”**命令。

有关详细信息，请参见《Vivado 设计套件用户指南：使用 IP 进行设计》 （参考文献 3] 和《*Vivado 设计套件用户指南：入门* （UG910） [参考文献 4]。

***注：***本章中的图为Vivado IntegratedDesign Environm Environm （IDE）的插图。此处描述的布局可能与当前版本不同。

*第 4 章：* **设计流程步骤**

### 输出生成

有关详细信息，请参见 *Vivado 设计套件用户指南：使用 IP 进行设计* （UG896） [参考文献 3]。

## 模拟

有关 Vivado 仿真组件的全面信息，以及有关使用受支持的第三方工具的信息，请参见 *Vivado 设计套件用户指南：逻辑仿真* （UG900） [参考文献 5]。

**重要说明：** *对于面向 Xilinx 7 系列 FPGA 或 Zynq-7000 器件的内核，不支持 UNIFAST 库。Xilinx IP 仅通过 UNISIM 库进行测试和认证。*

## 综合与实施

有关综合和实现的详细信息，请参阅 *Vivado 设计套件用户指南：*

*使用 IP 进行设计* （UG896） [参考文献 3]。

*第5章*

# 示例设计

本章包含有关 ERNIC 核心示例设计的信息。示例设计由以下模块组成：

* 时钟和复位发生器（仅仿真）
* 寄存器配置模块
* 发送、读取响应和 ACK 生成器
* 水质检测发生器
* TX Path Checker
* RX 路径检查器

图 5-1 显示了顶级示例设计体系结构。

X-参考目标 - 图 5-1

WQE 处理器

垂直标题

二甲基亚胺

布拉姆

阅读西格里

QP经理

配置

寄存 器

响应处理程序

RX 数据包处理程序

CRC 计算器

记忆

胶囊朝向

[

NVMof]

TX 数据包

检查

内存 [数据]

写

完成

读

王秦

包

发电机

]

TX-WQE

[

注册配置

模块

平方皮

更新

开始

文件

数据

对

NVMof

接收检查器

数据/门铃

数据包生成器

]

接收

[

读

从

记忆

记忆

标头和数据

]

[

传说

示例设计模块

XRNIC IP 模块

AXI4-流 I/F

AXI4 I/F

AXI4-精简版 I/F

边带 I/F

X23011-120220

*图 5-1：* **示例设计体系结构**

除了 ERNIC IP 之外，示例设计还集成了以下模块：

* **寄存器配置**模块：该模块配置 ERNIC IP 的所有必需寄存器。
* **数据包生成器模块：**这将生成以下类型的数据包：

° 发送数据包

来自 ERNIC IP 的所有读取请求的 °RDMA 读取响应数据包

来自 ERNIC IP 的所有 RDMA 写入请求的 °ACK 数据包

° RDMA 读取和 RDMA 写入

* **示例设计功能**：RDMA 读取和 RDMA 写入
* **RX 检查器模块**：此模块检查从 ERNIC IP 接收的胶囊（来自 SEND 数据包）和有效负载（来自读取响应）。该模块还检查 RQ 边带接口上响起的门铃数量。读取响应数据包、RDMA 写入重新任务的有效负载大小为256 字节。SEND 数据包中的胶囊大小为 80 字节。
* **WQE 生成器模块**：负责生成工作队列请求和 ERNIC IP 的 SQ PI 门铃更新。
* **TX 检查器模块**：检查 ERNIC IP 通过流接口传输的数据。
* **发起方 WQE 模块**：当发生 RDMA 读取时，这将从 DDR 获取数据。
* **启动器检查器模块**：

° 数据检查器，用于检查收到的 RDMA 写入操作的数据。

用于 RDMA REA D 操作的 °RDMA 读取响应数据包检查和针对 RDMA 写入操作的 ACK 数据包检查。

## 示例设计功能

示例设计演练了 ERNIC IP 的功能。以下传入数据包将发送到 ERNIC IP：

* RDMA 发送
* RDMA 读取响应
* 确认数据包
* RDMA READ
* RDMA 写入

示例设计检查以下传出数据包：

* RDMA Read Requests
* RDMA 写入请求
* RDMA 读取响应
* 确认数据包

以下是 ERNIC 的硬件握手路径：

* 仅执行 IPV4 数据包
* 仅行使六个 QP（QP2 至 QP7）

## 示例设计限制

ERNIC 示例设计具有以下限制：

* 示例设计不执行重试路径
* 仅示例设计练习 硬件握手路径（QPCONFi[4] 对所有 QP 设置为 0）
* 示例设计不执行 IPV6 数据包
* 示例设计不生成任何到 QP1 的连接管理 （MAD） 数据包
* 示例设计将每个受支持的数据包类型事务的数量限制为 8

## 模拟示例设计

有关仿真的更多信息，请参见 Vivado 设计套件用户指南：逻辑仿真 （UG900） [参考文献 5]。

### 仿真结果

仿真脚本编译 ERNIC 示例设计和支持仿真文件。然后，它会运行模拟并检查 以确保其成功完成。如果测试通过，则会显示以下消息：

**测试成功完成**

如果测试失败，则会显示以下消息：

**错误：测试失败**

如果测试挂起，则显示以下消息：

**错误：测试未完成（超时）**

## 示例序列

演示测试台执行以下任务：

* 所有必需的 ERNIC 寄存器都由寄存器配置 模块通过 AXI4-Lite 接口进行配置。
* 示例测试台中处理三个操作：
  1. RDMA 发送：
     + 数据包生成器模块生成 8 个发送到 QP2 到 QP7 的 SEND 数据包。
     + 示例设计的RX路径检查器，检查从ERNIC传输的胶囊上的数据完整性，同时门铃响起。
  2. RDMA RD 请求：
     + 示例设计具有预定义的 RDMA 读取工作队列条目数据包。WQE 生成器模块按响 SQPI 门铃，并在 ERNIC 请求时发送工作队列条目数据包。
     + TX 检查器模块检查收到的 RDMA 读取请求的必要字段。
     + 成功验证请求后，数据包生成器模块将 RDMA 读取响应发送到 ERNIC IP。
     + RX 路径检查器检查 ERNIC 传输的有效载荷的数据完整性，同时门铃响起。
  3. RDMA 写入请求：
     + 示例设计具有预定义的 RDMA 写入工作队列条目数据包。WQE 生成器模块按响 SQPI 门铃，并在 ERNIC 请求时发送工作队列条目数据包。
     + TX 检查器模块检查收到的 RDMA 写入 r 查询的必要字段。
     + 成功验证请求后，数据包生成器模块将 ACK 数据包发送到 ERNIC IP。
  4. RDMA 写：
  + 对于发起方功能，RDMA 写入数据包将发送到 ERNIC 模块，该模块将有效负载数据写入 RX 路径，并在 TX 路径上接收确认（ACK 数据包）。
  + ACK 数据包和数据通过检查器模块进行检查。

e.RDMA 阅读：

* 对于启动器功能，RDMA 读取数据包被发送到 ERNIC 模块，该模块从 DDR 读取数据并在TX 路径上发送 r ead 响应。
* 读取响应数据包通过检查器模块进行检查。

*第6章*

# ERNIC SoftwareFl ow

ERNIC IP 应先初始化和配置 ，然后才能用于发送和接收 RDMA 流量。这涉及按顺序执行以下步骤：

1. ERNIC IP 全局配置。
2. 内存区域注册（用于与远程主机共享内存区域的 RDMA 客户端应用程序）。
3. 创建用于发送和接收 RDMA 连接管理 （CM） 数据包的 QP1。
4. RC QP 创建。
5. CQ 创建。
6. 发布 RDMA 操作的 WQE。
7. 处理完成。
8. QP 删除。
9. QP致命恢复。

## ERNIC 全局配置

应为 ERNIC 全局配置配置以下寄存器。

* 配置错误缓冲区队列。分配内存并将基址配置为 ERRBUFBA 并将队列深度和大小配置为 ERRBUFSZ 寄存器。寄存器

硬件使用 ERRBUFWPTR 来指示有关 ERROR 缓冲区队列中新条目的软件。

* 配置传入数据包错误状态队列。 此队列提供传入数据包的状态，但有错误。要配置此队列，请分配内存并将基址写入 IPKTERRQBA，将队列深度和大小写入 IPKTERRQSZ 寄存器。

硬件使用 IPKTERRQWPTR 寄存器来指示有关队列中新条目的软件。

* 配置响应错误数据包缓冲区。这涉及将内存基址写入RESP\_ERR\_PKT\_BUF\_BA，将队列深度和大小写入RESP\_ERR\_BUF\_SZ，以及为硬件RESP\_ERR\_BUF\_WRPTR DDR地址，以指示SW关于发生的响应错误缓冲区写入。
* 通过写入g 到 INTEN 寄存器启用中断。
* 为 CQ 和 RQ 门铃分配内存，以便创建所有 QP。单个 CQ 和 RQ 的门铃存储器取自该存储器的偏移量，并在 QP 创建期间在相应的 ERNIC 寄存器中进行配置（参见 QP1 创建）。
* 在 MACXADDLSB 中配置源 MAC 地址，MACXADDMSB 寄存器。
* 配置源 IP 地址。如果 IP 版本为 IPv6，则寄存器 IPv6XADD1-4 将使用接口的 IPv6 地址写入，否则 IPv4 地址将写入 IPv4XADD 寄存器。
* 通过写入 XRNICCONF 寄存器来配置 ERNIC 设计、UDP 端口和 enab 的 ERNIC 支持的 QP 数量。

## QP1 创建

QP1是ERNIC IP中的特殊QP。此 QP 指定用于与远程主机交换 MAD 数据包，以便为 RC QP 建立连接。在创建任何其他 RC QP 之前，必须先配置 QP1。有关 QP1 的更多信息，请参阅 *InfiniBand 架构规范第* 1 卷、附录 A16 RoCE 和附录 17 RoCE V2 [参考文献 1]。配置和启用 QP1 需要执行以下步骤：

1. 为队列（RQ、SQ、CQ）分配内存 ，并对相应的基址寄存器进行编程。

° RQ 基址写入 RQBAi 寄存器。SQ 基址写入 SQBAi 寄存器，将队列深度写入 QDEPTHi。

° CQ 基址写入 CQBAi 寄存器。

° 通过写入 QPCONFi 来配置每个 RQ 缓冲区大小。

1. 作为 ERNIC 初始化的一部分，从预分配的门铃内存池中选取 SQ 完成门铃和 RQ 写入指针的内存（请参阅 ERNIC 全局配置）

° SQ完成门铃地址到CQDBADDi寄存器。

° RQ write pointer doorbell address to RQWPTRDBADDi register.

## 内存注册

内存必须先向 ERNIC 硬件注册，然后才能与远程主机交换以执行 RDMA 操作（传入 RDMA 读取和传入 RDMA 写入）。以下步骤对于向 ERNIC 注册内存非常重要：

1. 分配物理内存，并创建映射到它的虚拟地址。
2. 创建保护域号，并在 PD 表中的空闲插槽写入 PDPDNUM 寄存器。
3. 将 VIRTADDRLSB 中内存区域的虚拟地址写入 PD 表中的同一插槽和 VIRTADDRMSB 寄存器。
4. 将内存区域的物理地址写入 BUFBASEADDRLSB 和 BUFBASEADDRMSB 寄存器。
5. 创建一个R\_KEY并将其配置为缓冲区密钥寄存器。
6. 将内存区域长度写入 WRRDBUFLEN 寄存器，并将访问权限（远程读取或写入）写入 ACCESSDESC 寄存器。

## 遥控 QP 创建

启用 QP1 后，可以通过交换 QP1 上的 CM MAD 数据包来创建 RC QP。有关CM MAD包ETS的详细信息，请参阅*InfiniBand架构规范*第1卷，附录A16 RoCE和附录17 RoCE V2 [参考文献1]中的“通信管理”章节。要创建任何 RC QP，需要执行以下步骤：

1. 为队列（RQ、SQ、CQ）分配内存，并对相应的基址寄存器进行编程。

° RQ 基址写入 RQBAi 寄存器，SQ 基址写入 SQBAi 寄存器， 队列深度写入 QDEPTHi。

° CQ 基址写入 CQBAi 寄存器。

° 通过写入 QPCONFi 来配置每个 RQ 缓冲区大小。

1. 从预分配的门铃内存池中获取 SQ 完成门铃和 RQ 写入指针的内存，作为ERNIC 初始化的 pa rt。

° SQ完成门铃地址到CQDBADDi寄存器。

° RQ 将指针门铃地址写入 RQWPTRDBADDi 寄存器。

1. 当连接建立过程中 QP 状态从 RESET 转换为 RTS 时，请配置以下信息：
   1. 将远程主机 MAC 地址写入 MACDESADDLSBi 和 MACDESADDMSBi 寄存器。
   2. 使用目标 QP ID 配置 DESTQPCONFi。
   3. 将 IP 地址配置为 IPDESADDR1i - IPDESADDR4i 如果以其他方式使用 IPv6，请仅写入具有 IPv4 地址的 IPDESADDR1i。在 QPCONFi 中配置 IP 地址版本。
   4. 通过写入 SQPSNi 来配置发送队列 PSN。
   5. 通过写入 LSTRQREQi 来配置最后一个接收队列 PSN
   6. 通过写入 QPCONFi 寄存器来配置 PMTU。
   7. 通过写入 TIMEOUTCONFi 寄存器来配置以下内容。
      * 配置 RNR 重试计数和 RNR 重试超时
      * 配置重试超时和重试超时
   8. 与QP关联的PD号码通过写入保护域号码寄存器来共同计算。

## WQE SQ 发布

* QP 现在可用于发布 RDMA 操作的工作请求。为此，请按表 2-1 中所述的格式准备 WQE 并将其复制到 SQ 内存。
* 通过将 SQPIi 递增 1 并将值写入 SQPIi 寄存器来按门铃。

## 接收传入的 RDMA 发送消息

传入的 SEND 消息中接收的数据由 ERNIC IP 复制到 RQ 缓冲区。该软件可以通过执行以下步骤来检索此数据：

1. 读取 RQWPTRDBADDi 值。如果与以前的值有变化，则通过从 RQBAi+offset 读取数据来处理接收到的数据。此偏移量是根据 RQ 缓冲区大小和 RQWPTRDBADDi 中的值计算的。
2. 处理收到的消息后，递增 RQCIi 寄存器值，以向硬件指示缓冲区已消耗，以便可用于更多传入消息。

## 处理 WQE 完井

ERNIC IP 通过递增 CQHEADi 寄存器来指示发布到它的 WQE 的完成。软件应轮询此寄存器，以检查以前发布的 WQE 是否有任何完成情况。

## Enabling QP HW 卸载

RC QP 可以启用硬件握手模式，在该模式下，外部硬件应用程序可以在此 QP 上发送和接收 RDMA 消息。将 RDMA 消息的软件发布卸载到外部硬件应用程序。要在 QP 上启用硬件 handshake 模式，需要执行以下步骤：

1. 通过写入XRNIC\_ADV\_CONF启用软件覆盖。
2. 通过将 0 写入 SQPIi、CQHEADi 和 STATCURSQPTRi 来重置 QP 指针。
3. 通过获取 RQCIi 和 RQWPTRDBADDi 之间的差值来配置要处理的剩余 RQ，并将值写入较低 2B 的 STATRQPIDBi。
4. 通过写入 QPCONFi 启用硬件卸载。
5. 通过写入XRNIC\_ADV\_CONF来禁用软件覆盖。

## QP 删除

当 RC 不再使用时（由于本地启动的连接或与远程对等体的连接），可以将其删除。删除 QP 并删除其配置需要执行以下步骤：

1. 等待 SQ 和未完成的队列为空。状态位位于 STATQPi 寄存器中。
2. 等待 SQ 中收到的 WQE 的所有完成。这是通过检查 SQPIi 和 CQHEADi 寄存器来完成的。
3. 通过写入XRNIC\_ADV\_CONF寄存器来启用软件覆盖，并通过写入t o QPCONFi 寄存器来禁用 QP。
4. 通过将 0 写入 RQWPTRDBADDi、SQPIi、CQHEADi、RQCIi、STATRQPIDBi、STATCURSQPTRi、SQPSNi、LSTRQREQi、STATMSN 来重置 QP 指针，然后通过写入 QPCONFi 禁用 QP 并将其保持在恢复模式。
5. 重置指针后，应通过写入XRNIC\_ADV\_CONF来禁用 softwa 重新覆盖。
6. 在此之后，软件应释放分配给 SQ、RQ 和 CQ 的内存。

## QP 致命恢复

以下步骤描述了进入致命状态的 QP 的恢复过程。它描述了如何清除 QP 上的现有流量并重新初始化它，以便可以重复使用。

清除 QP 上的流量的步骤：

1. 在致命中断下检测到 QP 时，请阅读 IPKTERRQBA 寄存器以了解驱动程序写入的传入 Pkt 错误状态队列基址。
2. 阅读此基址以了解 QP 致命状态，并确定哪个 QP 进入致命状态 （Bits[31：16]） 并检查 QP 致命状态代码 （Bits[15：0]），请参阅 表 2-4。
3. 停止再按任何 SQ PI 门铃。
4. 将 QPCONFi 寄存器中的“正在恢复的 QP”位设置为 1。
5. 读取 STATQPi 寄存器以检查“发送 Q 空”和“未完成 Q 空”位变为 1。
6. 轮询 CQHEADi 寄存器以检查其值是否与 SQPIi 寄存器相同。
7. 轮询 RESPHNDSTS 寄存器以查找 “sq pici db check en” — 要设置的第 16 位。
8. 将 QPCONFi 寄存器“QP 启用”位设置为 0，将“正在恢复的 QP”位设置为 1。

重新初始化QP 的步骤：

1. 轮询 CQHEADi 寄存器以检查其值是否与 SQPIi 寄存器相同。
2. 轮询 RESPHNDSTS 寄存器以查找 “sq pici db check en” — 要设置的第 16 位。
3. 将XRNICADCONF寄存器中的“SW OVERWRIDE”位设置为1。
4. 将以下 QP 寄存器初始化 为 0：

° STATRQPIDBi

° STATRQBUFCAi

° STATRQBUFCAMSBi

° RQCIi

° STATCURSQPTRi

° SQPIi

° SQPSNi

° LSTRQREQi

° 统计局

° CQHEADi

1. 轮询 CQHEADi 寄存器以检查其值是否为 0。
2. 使用新值初始化以下寄存器：

° SQPSNi

° LSTRQREQi

1. 初始化以下以太网端寄存器：

° MACDESADDMSBi

° MACDESADDLSBi

° IPDESADDR1i

° IPDESADDR2i

° IPDESADDR3i

° IPDESADDR4i

1. 在 QPCONFi 中重新配置 IP 版本。
2. 重新初始化 STATQPi 寄存器中的“RNR nack count”和“retry count”。
3. 将该 QP 的 ACCESSDESC 寄存器“访问类型”设置为 **'b10**。
4. 通过重新初始化“RQ interrupt enable”、“CQ 中断启用”、“PMTU”和“HW 握手禁用”等字段，重新编程 QPCONFi 寄存器。有选择地启用“CQE 写入启用”以调试错误完成并重新初始化“RQ 缓冲区大小”。
5. 通过随机化“流量类”、“生存时间”和“分区键”来重新编程 QPADVCONFi 寄存器。
6. 在 QPCONFi 寄存器中将“QP EN”设置为 1，将“正在恢复的 QP”设置为 0。
7. 将XRNICADCONF寄存器中的“'SW OVERWRIDE”位清除为0。

*附录A*

# 请求 ERNIC 支持问卷

以下是在请求 ERNIC 支持之前要填写的调查问卷。

*表 A-1：* **ERNIC 支持问卷**

|  |  |  |
| --- | --- | --- |
| **问题** |  | **答** |
| **ERNIC IP 配置** | |  |
| ERNIC 和 Vivado 发布版本 |  |  |
| 硬件 GUI 参数值 |  |  |
| 启用/禁用硬件握手模式 |  |  |
| **ERNIC 系统级连接** | |  |
| 厄尼克 厄尼克 ↔ |  |  |
| 厄尼克开关 ↔ 厄尼克 ↔ |  |  |
| 厄尼克↔·梅拉诺克斯 |  |  |
| ERNIC ↔ 开关 ↔ 梅拉诺克斯 |  |  |
| ERNIC ↔ BFM （模拟） |  |  |
| **流量模式** | |  |
| 操作码 – RDMA\_READ/RDMA\_WRITE/RDMA\_SEND/  立即/无效/混合具有不同操作码的数据包 |  |  |
| 传输大小（DMA 长度和 PMTU） |  |  |
| **交通方向** | |  |
| ERNIC 发送数据包 （TX） |  |  |
| ERNIC 接收数据包 （RX） |  |  |
| 双 |  |  |
| **问题类型** | |  |
| 挂  ERNIC 未发送确认  ERNIC 未处理 WQE  ERNIC 未处理传入的请求/响应 |  |  |
| 性能下降 |  |  |
| 丢包 |  |  |
| 重试次数过多 |  |  |

*附录 A：* **请求 ERNIC 支持问卷**

*表 A-1：* **ERNIC 支持问卷 *（续）***

|  |  |  |
| --- | --- | --- |
| **问题** |  | **答** |
| 意外致命 – 检查主机是否发送了致命响应 |  |  |
| 致命的恢复 |  |  |
| FCS 错误 – 检查数据包 FIFO 在 之间的行为  ERNIC → CMAC |  |  |
| 其他 |  |  |
| **拥塞** | |  |
| PFC 在 ERNIC 和 CMAC 中启用？ |  |  |
| CMAC 的 PFC 计数器值是多少？ |  |  |
| **软件相关问题** | |  |
| 赛灵思提供的参考设计存在问题？ |  |  |
| 连接建立是通过 MAD 数据包完成的？ |  |  |
| **其他** | |  |
| 系统框图 |  |  |
| 连接到 ERNIC 的任何自定义块的详细信息 |  |  |
| 注册配置详细信息  配置寄存器转储  调试寄存器转储 |  |  |
| 模拟/线鲨转储 |  |  |

*附录B*

# 调试

本附录包含有关 Xilinx 支持网站和调试工具上可用资源的详细信息。在联系 Xilinx 支持部门之前，请填写附录 A 中提供的基本问卷。

**提示：***如果 IP 生成因错误而停止，则此处可能是许可证问题。有关 更多详细信息，*请参阅*第 1 章中的许可证检查器*。

## 调试 IP/系统

ERNIC IP 具有许多状态寄存器，以便在发生故障时进行广泛的调试。这些寄存器以及来自启动器RNIC（例如Mellanox）的计数器信息可以为调试任何故障提供重要线索。使用 ERNIC IP 的系统调试器可用的 deb ug 工具包括：

* ERNIC 完全寄存器转储
* 混杂或数据包嗅探器模式计数器（来自发起方 RNIC），
* 硬件计数器（来自启动器 RNIC）。

以下各节介绍了转储不同日志的步骤。

### 完全转储 ERNIC 寄存器

如果您使用的是智能电缆，请执行以下步骤：

1. 使用以下命令将连接命令指令更改为智能电缆的相应 IP 地址：

**$ connect -host <smartlink IP address>**

1. 使用以下命令更改目标：

**目标 -设置 -nocase -filter {name =~ “<处理器名称>”}**

1. 使用以下命令读取 ERNIC IP 寄存器值：

**$ mrd - 强制<ERNIC IP 偏移量> <要读取的位置数>**

### 在启动器 RNIC 上启用杂项模式

1. 使用以下命令在启动器 RNIC 上启用杂项模式：

**$ ifconfig <接口名称> promisc**

**$ netstat -i**

1. 在测试结束时，使用以下命令从启动器 RNIC 转储所有计数器信息：

**$ ethtool –S <RNIC 卡名> > ethtool.log**

### 美兰诺 RNIC 的硬件调试计数器

要访问和转储 Mellanox RNIC 的硬件调试计数器，请使用以下命令：

**$ cd /sys/class/infiniband/mlx5\_0/ports/1/hw\_counters/**

**$ 表示 'ls .' 中的文件 .;do echo -n “${file}：”;猫$file;做**

***注意：***[有关 Mellanox 硬件调试计数器的详细信息，请参阅 Mellanox 社区上的 DOC-2572](https://community.mellanox.com/docs/DOC-2572.)。

此处列出了一些可以执行的快速调试检查，以确保系统干净。

• 检查 ethtool.log 文件中是否有任何链路故障或 CRC 故障。这两个计数器中的任何非零值都指向不稳定的链接，并且可能是失败的原因。 此处列出了 **ethtool.log** 文件中的两个片段。

示例 1：

**rx\_wqe\_err： 0**

**rx\_mpwqe\_filler：0 rx\_mpwqe\_frag：0 rx\_buff\_alloc\_err：0 rx\_cqe\_compress\_blks：0 rx\_cqe\_compress\_pkts：0 link\_down\_events\_phy：4 rx\_out\_of\_buffer：0 rx\_vport\_unicast\_packets：5 rx\_vport\_unicast\_bytes：422 Tx\_vport\_unicast\_packets：10 tx\_vport\_unicast\_bytes：714 rx\_vport\_multicast\_packets： 46 rx\_vport\_multicast\_bytes：7608** 样本 2：

**rx\_vport\_rdma\_multicast\_bytes： 0**

**tx\_vport\_rdma\_multicast\_packets：0 tx\_vport\_rdma\_multicast\_bytes：0 tx\_packets\_phy：320587336 rx\_packets\_phy：324924215 rx\_crc\_errors\_phy：0 tx\_bytes\_phy：27657272230 rx\_bytes\_phy：467673988652 tx\_multicast\_phy ：59 tx\_broadcast\_phy：32 rx\_multicast\_phy：0 rx\_broadcast\_phy：9 rx\_in\_range\_len\_errors\_phy：0**

* 检查启动器端的hw\_counters。这些计数器提供了发起方看到的所有致命/非致命错误的图片。计数器示例：

**duplicate\_request：0implied\_nak\_seq\_err：0寿命：10 local\_ack\_timeout\_err：0out\_of\_buffer：0out\_of\_sequence：0packet\_seq\_err：0rnr\_nak\_retry\_err：0rx\_atomic\_requests：0rx\_read\_requests：5 rx\_write\_requests：108307999**

* 从 ERNIC 寄存器转储中检查以下寄存器位置。所有已启用的 QP 的 QP 状态 （STATQPi） 寄存器提供不同 QP 的状态。检查任何 QP 状态寄存器中的 QP 致命状态是否设置为 1。例如，QP 5 的 QP 状态寄存器：

**0x84020688： 30620601 ？QP 致命设置为 1**

* 如果 QP 处于致命状态，则不会从此 QP 执行任何事务，并且 QP 将断开连接。同一寄存器中的位[22：16]提供从发起者收到的最后一个AETH综合征。在许多卡斯中，由于从启动者那里接受的NAK综合征，QP可能会进入致命状态。NAK 综合征可帮助您了解启动器 RNIC 卡看到的故障。在上面的示例中，0x62 AETH 综合征表示来自发起方的“远程访问错误”。AETH 综合征的解码在 [Infiniband 架构规范第 1 卷（版本 1.2.1）](http://www.afs.enea.it/asantoro/V1r1_2_1.Release_12062007.pdf)中提供[。](http://www.afs.enea.it/asantoro/V1r1_2_1.Release_12062007.pdf) 有关此 s 特殊情况中的 NAK 代码详细信息，请参阅表 43：AETH 综合征字段和表 44：NAK 代码。
* 在偏移**0x134**和**0x138**处检查传入和传出的NAK计数寄存器（（INNACKPKTCNT）和（OUTNACKPKTCNT）），以了解观察到的传入NAK综合征的数量和发出的NAK合成罗马的数量。此数字通常应与从启动器看到**的hw\_counters**中看到的数字相关。一般来说，并非所有NAK代码都是致命的。但是，所有 NAK 代码都会导致重试，并可能降低系统的整体性能。NAK代码的数量可能令人担忧。
* 目标启动的重试总数可以从偏移**量0x140**的重试计数状态寄存器 （RETRYCNTSTS） 中知道。通常，此数字将与传入的 NAK 计数匹配。如果此数字大于传入的 NAK 计数值，则可能是由于超时。当响应方（在本例中为发起方 RNIC）在给定时间内未响应请求时，会发生超时。超时值在超时配置寄存器 （TIMEOUTCONF） 中配置。此超时间隔根据 [InfiniBand™ 架构规范](http://www.afs.enea.it/asantoro/V1r1_2_1.Release_12062007.pdf)第 1 卷[（版本 1.2.1）](http://www.afs.enea.it/asantoro/V1r1_2_1.Release_12062007.pdf)条款 C9-141 实现。尝试增加超时间隔并检查重试次数是否减少可能是值得的。
* ERNIC 寄存器**0x6C** （ERRBUFWPTR） 表示错误缓冲区写入指针。此寄存器给出收到的错误数据包数。每个错误数据包将存储在错误缓冲区基址 （ERRBUFBA） 寄存器（偏移量0x60）中提供的地址位置。 此缓冲区中的每个条目都将带有错误综合症。有关详细信息，请参阅 ERNIC RX 路径。以黄色突出显示的行列出了将导致 QP 进入致命状态的条件。

*附录C*

# 其他资源和法律声明

## 赛灵思资源

有关答案、文档、下载和论坛等支持资源，请参阅 [Xilinx 支持](https://www.xilinx.com/support)。

有关 Xilinx 文档中使用的技术术语表，请参见 Xilinx 术语表。

## 文档导航器和设计中心

Xilinx 文档导航器 p提供对 Xilinx® 文档、视频和支持的访问

资源，您可以对其进行筛选和搜索以查找信息。要打开 Xilinx 文档导航器 （DocNav）：

* 从 Vivado® IDE 中，选择“ **帮助”>“文档和教程**”。
* 在Windows上，选择启动**>所有程序>Xilinx设计工具>DocNav**。
* 在 Linux 命令提示符下，输入 **docnav**。

Xilinx 设计中枢提供按设计任务和其他主题组织的文档链接，您可以使用这些链接来学习关键概念并解决常见问题。访问设计中心：

* 在 Xilinx 文档导航器中，单击 **设计 中心视图** 选项卡。
* 在 Xilinx 网站上，请参阅 [设计中心](https://www.xilinx.com/cgi-bin/docs/ndoc?t=design+hubs) 页面。

***注意：***有关文档导航器的更多信息，请参阅 Xilinx 网站上的[文档导航器](https://www.xilinx.com/cgi-bin/docs/rdoc?t=docnav)页面。

## 引用

这些文档提供了对本产品指南有用的补充材料：

1. *InfiniBand 架构规范第* [1 卷，附录 A16 RoCE 和附录 17 RoCE V2](https://www.infinibandta.org/ibta-specifications-download/)
2. *Vivado 设计套件用户指南：使用 IP 集成器设计 IP* 组件 ems （[UG994）](https://www.xilinx.com/cgi-bin/docs/rdoc?v=latest;d=ug994-vivado-ip-subsystems.pdf)
3. *Vivado 设计套件用户指南： 利用 IP 进行设计* （[UG896）](https://www.xilinx.com/cgi-bin/docs/rdoc?v=latest;d=ug896-vivado-ip.pdf)
4. *Vivado 设计套件用户指南：入门* （[UG910）](https://www.xilinx.com/cgi-bin/docs/rdoc?v=latest;d=ug910-vivado-getting-started.pdf)
5. *Vivado 设计套件用户指南：逻辑仿真* （[UG900）](https://www.xilinx.com/cgi-bin/docs/rdoc?v=latest;d=ug900-vivado-logic-simulation.pdf)
6. *Vivado 设计套件用户指南：编程和调试* （[UG908）](https://www.xilinx.com/cgi-bin/docs/rdoc?v=latest;d=ug908-vivado-programming-debugging.pdf)
7. *Vivado 设计套件用户指南：实施* （[UG904）](https://www.xilinx.com/cgi-bin/docs/rdoc?v=latest;d=ug904-vivado-implementation.pdf)
8. *Vivado 设计套件用户指南：* AXI 参考指南 （[UG1037）](https://www.xilinx.com/cgi-bin/docs/ipdoc?c=axi_ref_guide;v=latest;d=ug1037-vivado-axi-reference-guide.pdf)

## 修订历史

下表显示了此文档的修订历史记录。

|  |  |  |
| --- | --- | --- |
| **日期** | **版本** | **校订** |
| 06/30/2021 | 3.1 | 在表 3-1 中将大小更新为 256。 |
| 06/16/2021 | 3.1 | * 添加了 Versal ACAP。 * 在功能中添加了计时冲突说明。 * 更新了 图 1-1。 * 在 RX PKT 处理程序中添加了非 RDMA 数据包描述。 * 在响应处理程序中添加了重新传输说明。 * 更新了最大 DMA 长度，并在不支持的功能中添加了读取响应说明。 * 更新了 表 2-4。 * 在表 2-5 中添加了 rx\_pkt\_hndler\_ddr\_m\_axi\* 的说明。 * 更新了表 2-7 中 0x20088、0x2013C、0x20200 + （（i-1） x 0x0100） 和 0x202B0 + （（i-1） x 0x0100） 的说明。 * 更新了表 3-1 中的大小。 * 增加了 QP致命恢复。 * 添加了 附录 A，请求 ERNIC 支持问卷。 |
| 02/26/2021 | 3.0 | 更新了“资源利用率”链接。 |

|  |  |  |
| --- | --- | --- |
| **日期** | **版本** | **校订** |
| 01/21/2021 | 3.0 | * 更新了功能中的优先级流控制。 * 添加了按设计流程导航内容部分。 * 在功能摘要中添加了流控制管理器。 * 更新了 图 1-1。 * 在 ERNIC RX 路径中添加了 RDMA 读/写请求描述。 * 更新了 图 2-6。 * 更新了表 2-5 中的 CMAC RX/TX 描述和 RoCE 描述。 * 更新了 图 2-9。 * 更新了 表 2-7。 * 更新了 图 3-1。 * 更新了 Rcv Q 缓冲区基址 QPi （RQBAMSBi） 的地址。 * 更新了 ERNIC 寄存器完全转储中的描述 #2 和 #3。 |
| 06/10/2020 | 2.0 | 添加了新章节 第 6 章 ERNIC 软件流程 |
| 12/09/2019 | 1.0 | 更新了表 2-5 和表 3-2。 |
| 07/10/2019 | 1.0 | 次要更新 |
| 12/05/2018 | 1.0 | 初始赛灵思版本。 |

## 请阅读：重要法律声明

本协议向您披露的信息（以下简称“材料”）仅用于选择 Xilinx 产品。在适用法律允许的最大范围内：（1） 材料按“原样”提供，Xilinx 特此声明所有明示、暗示或法定的保证和条件，包括但不限于对适销性、非侵权性或适用于任何特定用途的保证; （2） Xilinx 不承担责任（无论是合同还是侵权行为， 包括疏忽，或根据任何其他责任理论）与材料（包括您对材料的使用）相关、由材料引起或与之相关的任何种类或性质的任何损失或损害，包括任何直接、间接、特殊、间接或后果性损失或损害（包括数据、利润、商誉的损失或因第三方提起的任何诉讼而遭受的任何类型的损失或损害） 甚至如果此类损坏或损失是可合理预见的，或者 Xilinx 已被告知发生此类损害或损失的可能性。Xilinx 没有义务更正材料中包含的任何错误，也没有义务通知您材料或产品规格的更新。未经事先书面同意，您不得复制、修改、分发或公开展示 Mate里亚尔。某些产品受 Xilinx 有限保修条款和条件的约束，请参阅 Xilinx 的销售条款，可在 <https://www.xilinx.com/legal.htm#tos> 查看[;](https://www.xilinx.com/legal.htm#tos) IP 核可能受 Xilinx 颁发给您的许可证中包含的保修和支持条款的约束。Xilinx 产品并非设计或旨在实现故障安全，也不适用于任何需要故障安全性能的应用;在此类关键应用中使用 Xilinx 产品，您承担全部风险和责任，请参阅 Xilinx 的销售条款，可在 <https://www.xilinx.com/legal.htm#tos> 查看[。](https://www.xilinx.com/legal.htm#tos)

**汽车应用免责声明**

汽车产品（在部件号中标识为“XA”）不保证用于部署

安全气囊或用于影响车辆控制的应用（“安全应用”），除非

符合 ISO 26262 汽车安全标准（“安全”）的安全概念或冗余功能

设计“）。在使用或分发任何包含产品的系统之前，客户应彻底

出于安全目的测试此类系统。 在没有安全设计的情况下在安全应用中使用产品完全由客户承担风险，仅受有关产品限制的适用法律和法规的约束

责任。

© 版权所有 2018-2021 Xilinx， Inc. Xilinx、Xilinx 徽标、Alveo、Artix、Kintex、Spartan、Versal、Virtex、Vivado、Zynq 以及此处包含的其他指定品牌均为 Xilinx 在美国和其他国家/地区的商标。AMBA、AMBA Designer、Arm、ARM1176JZ-S、CoreSight、Cortex、PrimeCell、Mali 和 MPCore 是 Arm Limited 在欧盟和其他国家/地区的商标。DisplayPort 图标是视频电子标准协会的商标，在美国和其他国家/地区注册。所有其他商标均为其各自所有者的财产。