**AXI DMA v7.1**

***罗技核知识产权产品指南***

**Vivado 设计套件**

# **PG021 4月 27， 2022**

Xilinx 正在营造一个让员工、客户和合作伙伴感到宾至如归和包容的环境。为此，我们将从我们的产品和相关宣传资料中删除非包容性语言。我们发起了一项内部计划，旨在删除 可能排除人员或强化历史偏见的语言，包括嵌入在我们的软件和 IP 中的术语。您可能仍然可以在我们的旧产品中找到非包容性语言的示例，因为我们正在努力进行这些更改并与不断发展的行业标准保持一致。 点击此 [链接](https://www.xilinx.com/content/dam/xilinx/publications/about/Inclusive-terminology.pdf) 了解更多信息。



# 目录

**知识产权事实**

## 第 1 章：概述

**功能摘要 . 6 申请 . . . . . . . . . . . . . . . . . . . . . . . . 7**

**许可和订购 . 7**

## 第2章 产品规格

**性能。。。。。。。 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 8 资源利用 . . . . . . . . . 10 港口说明 .. 10 注册空间 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 11 分散收集描述符 .. 38**

**多通道 DMA 支持 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 56**

## 第 3 章：使用核心进行设计

**典型系统互连 . . . . . . . . . . . . 68 时钟 . 69 重置 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 70**

**编程顺序 . 70**

## 第4章 设计流程

**定制和生成核心 . 75 约束核心 . . . . . . . . . . . . . . . 83 模拟 . 84**

**综合与实施 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 84**

## 第 5 章：示例设计

**实施示例设计 .. 87 模拟示例设计 . . . . . . . . . . 88**

**示例设计的试验台 . 89**

发送反馈

## 附录 A：升级

**迁移到 Vivado 设计套件 . 90**

**在 Vivado 设计套件中进行升级 . 90**

## 附录 B：调试

**寻求 Xilinx.com 帮助 . . . . . . . . . . . . . . . . . . . . . . . 91 Vivado 设计套件调试功能 . 92**

**硬件调试 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 93**

## 附录 C：其他资源和法律声明

**赛灵思资源 .. . . . . . . . . . . . 94 文档导航器和设计中心 . 94 参考资料 .. . . . . . . . . . . . . . . . . . . . . . . . . . . 95 修订历史 . 95 请阅读：重要法律声明 . .** . . **. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 98**

## 介绍

**知识产权事实**



Xilinx® LogiCORE™ IP AXI Direct Memory Access （AXI DMA） 内核是一款软 Xilinx IP 内核，可与 Xilinx Vivado® 设计套件配合使用。AXI DMA 在内存和内存之间提供高带宽直接内存访问

AXI4 流目标外设。它的 选项l分散/收集功能还可以从中央处理器 （CPU） 卸载数据移动任务。

## 特征

* 符合 AXI4 标准
* 可选的分散/收集直接内存访问 （DMA） 支持
* AXI4 数据宽度支持 32、64、128、256、

512 位和 1，024 位

* AXI4 流数据宽度支持 8、16、

32、64、128、256、512 和 1，024 位

* 可选锁孔支架
* 可选的数据重新对齐支持，支持高达 512 位的流数据宽度
* 可选的 AXI 控制和状态流
* 可选的微型DMA支持
* 支持高达 64 位寻址

|  |  |
| --- | --- |
|  | **LogiCORE 知识产权事实表** |
|  | **核心细节** |
| 支持  装置  家庭（1） | Versal® ACAP， UltraScale+ ™  超大规模™  Zynq-7000® 芯片、赛灵思 7 系列 FPGA |
| 支持  用户界面 | AXI4， AXI4-Lite， AXI4-Stream |
| 资源 | [性能和资源利用率网页](https://www.xilinx.com/cgi-bin/docs/ndoc?t=ip+ru;d=axi-dma.html) |
|  | **提供核心** |
| 设计文件 | VHDL |
| 例  设计 | VHDL |
| 试验台 | VHDL |
| 约束  文件 | 通过 IP 生成提供 |
| 支持  软件驱动器（2） | 独立和 Linux |
|  | **经测试的设计流程（3）** |
| 设计条目 | Vivado 设计套件 |
| 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=2022.1;t=vivado+release+notes) |
| 合成 | 维瓦多合成 |
|  | **支持** |
|  | [赛灵思支持网页](https://www.xilinx.com/support) |

1. 有关受支持器件的完整列表，请参见 Vivado IP 目录。
2. 可以在软件开发工具包 （SDK） 安装目录中找到独立驱动程序信息。请参阅xilinx\_drivers.htm

<install\_directory>/SDK/<release>/data/embeddedsw/ doc/xilinx\_drivers.htm.

1. 有关这些工具的支持版本，请参见[Xilinx 设计套件：发行说明指南。](https://www.xilinx.com/cgi-bin/docs/rdoc?v=2022.1;t=vivado+release+notes)

发送反馈



*第一章*

# 概述

AXI 直接内存访问 （AXI DMA） IP 核在 AXI4 内存映射和 AXI4 流 IP 接口之间提供高带宽直接内存访问。其可选的分散收集功能还可以从 基于处理器的系统中的中央处理单元 （CPU） 卸载数据移动任务。初始化、状态和管理寄存器可通过 AXI4-Lite 从机接口访问。 图1-1 说明了内核的功能组成。

X-参考目标 - 图 1-1

数据移动器

寄存 器

MM2S Cntl/Sts Logic

S2MM Cntl/Sts Logic

数据移动器

分散/聚集

AXI4 内存映射读取

AXI4-精简版

AXI4 内存映射写入

AXI4 流从站 （S2MM）

AXI4 流 （S2MM）

AXI4 内存映射写入/读取

AXI4 控制流 （MM2S）

AXI4 流主机 （MM2S）

x13225

*图 1-1：* **AXI DMA 框图**

 *第 1 章：* **概述**

系统内存和流目标之间的主要高速 DMA 数据移动是通过 AXI4 读主服务器到 AXI4 内存映射到流 （MM2S） 主服务器，以及 AXI 流到内存映射 （S2MM） 从属到 AXI4 写主机。AXI DMA 还支持在分散/收集模式下在 MM2S 和 S2MM 路径上实现多达 16 个多通道的数据移动。

MM2S 通道和 S2MM 通道独立运行。AXI DMA 提供 4 KB 地址边界保护（在非微型 DMA 中配置时）、自动突发映射， 以及使用几乎 AXI4-Stream 总线的全部带宽功能对多个传输请求进行排队的能力。此外，AXI DMA 提供字节级数据重新对齐，允许从任何字节偏移位置开始进行内存读取和写入 。

MM2S 通道支持 AXI 控制流，用于将用户应用程序数据发送到目标 IP。对于 S2MM 通道，提供 AXI 状态流，用于从目标 IP 接收用户应用程序数据。

可选的分散/收集 引擎通过 AXI4 分散收集读/写主接口从系统内存中获取和更新缓冲区描述符。

## 功能摘要

* 符合 AXI4 标准
* 可选的独立分散/收集直接内存访问 （DMA） 支持

° 提供从 CPU 卸载 DMA 管理工作

° 提供独立于主数据总线的传输描述符的获取和更新

° 允许将描述符放置在与数据缓冲区分开的纽约内存映射位置。例如，描述符可以放置在块 RAM 中。

° 提供可选的循环操作

* 可选直接寄存器模式（不支持分散/聚集）

通过排除分散收集引擎，可以启用性能较低但 FPGA 资源密集程度较低的模式。在此模式下，通过设置源地址（对于 MM2S）或目标地址（对于 S2MM），然后在长度寄存器中指定字节计数来命令传输。

* 主 AXI4 数据宽度支持 32、64、128、256、512 和 1，024 位
* 主 AXI4 流数据宽度支持 8、16、32、64、128、256、512 和 1，024 位
* 可选的数据重新对齐引擎，适用于高达 512 位的流数据宽度

*第 1 章：* **概述**

所有数据都重新对齐到主存储器映射和流数据路径上的字节（8 位）级别

* 可选的 AXI 控制和状态流，用于连接 AXI 以太网 IP

为 MM2S 通道提供可选的控制流，为 S2MM chan nel 提供状态流，以从高带宽数据路径卸载低带宽控制和状态。

* 可选微模式

AXI DMA 可以配置为提供低占用空间、低性能的 IP，可以处理小数据包的传输。阅读以下章节以获取更多信息。

## 应用

AXI DMA 在系统内存和基于 AXI4 流的目标 IP（如 AXI 以太网）之间提供高速数据移动。

## 许可和订购

这款 Xilinx LogiCORE™ IP 模块根据 Xilinx [最终用户许可条款随 Xilinx](https://www.xilinx.com/cgi-bin/docs/rdoc?t=eula)® Vivado® 设计套件免费提供。

有关此模块和其他 Xilinx LogiCORE IP 模块的信息，请访问 [Xilinx](https://www.xilinx.com/products/intellectual-property.html)

[知识产权 页面。有关蓋毒素](https://www.xilinx.com/products/intellectual-property.html)g 以及其他 Xilinx LogiCORE IP 模块和工具可用性的信息，请联系您 [当地的 Xilinx 销售代表](https://www.xilinx.com/about/contact.html)。



*第2章* 产品规格

## 性能

本节包含以下小节。

* [性能](https://www.xilinx.com/htmldocs/ip_docs/pru_files/axi-dma.html)与资源利用率
* 延迟和吞吐量

### 延迟和吞吐量

表 2-1 和表 2-2 描述了 AXI DMA 的延迟和吞吐量。这些表提供了典型配置的性能信息。吞吐量测试包括在 MM2S 和 S2MM 端传输 10，000 个字节。

吞吐量是从完成描述符提取 （DMACR.空闲 = 1） 到帧计数中断断言。

*表 2-1：* **AXI DMA 延迟数字**

|  |  |
| --- | --- |
| **描述** | **时钟** |
| **MM2S 通道** |  |
| 尾部描述符写入m\_axi\_sg\_arvalid | 10 |
| m\_axi\_sg\_arvalid到m\_axi\_mm2s\_arvalid | 28 |
| m\_axi\_mm2s\_arvalid到m\_axis\_mm2s\_tvalid | 6 |
| **S2 毫米通道** |  |
| 尾部描述符写入m\_axi\_sg\_arvalid | 10 |
| s\_axis\_s2mm\_tvalid到m\_axi\_s2mm\_awvalid | 39 |

*表 2-2：* **AXI DMA 吞吐量数（1）**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **渠道** | **时钟频率（兆赫）** | **传输的字节数** | **总吞吐量（MB/秒）** | **理论百分比** |
| MM2S（2） | 100 | 10,000 | 399.04 | 99.76 |

*表 2-2：* **AXI DMA 吞吐量数（1）（ *续）***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **渠道** | **时钟频率（兆赫）** | **传输的字节数** | **总吞吐量（MB/秒）** | **理论百分比** |
| S2毫米（3） | 100 | 10,000 | 298.59 | 74.64 |

**笔记：**

1. 上述数字是使用默认 IP 配置测量的。
2. MM2S 吞吐量是在内存映射端的第一个 arvalid 与流式处理端的第一个 arvalid 之间测量的。
3. S2MM 吞吐量是在流式处理端的第一个 tvalid 到 内存映射端的最后一个 tvalid 之间测量的。

## 资源利用率

有关性能和资源利用率的完整详细信息，请访问 [性能和资源利用率网页。](https://www.xilinx.com/cgi-bin/docs/ndoc?t=ip+ru;d=axi-dma.html)

## 端口说明

AXI DMA I/O 信号如表 2-3 所示。

*表 2-3：* **I/O 信号 说明**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **信号名称** | **接口** | **信号**  **类型** | **初始化**  **地位** | **描述** |
| s\_axi\_lite\_aclk | 时钟 | 我 |  | AXI4 精简版时钟。 |
| m\_axi\_sg\_aclk | 时钟 | 我 |  | AXI DMA 散射聚集时钟 |
| m\_axi\_mm2s\_aclk | 时钟 | 我 |  | AXI DMA MM2S 主时钟 |
| m\_axi\_s2mm\_aclk | 时钟 | 我 |  | AXI DMA S2MM 主时钟 |
| axi\_resetn | 重置 | 我 |  | AXI DMA 重置。/有效复位。断位为低时，将重置整个 AXI DMA 内核。必须与s\_axi\_lite\_aclk同步。 |
| mm2s\_introut | 中断 | O | 0 | 中断输出以内存映射到流通道。 |
| s2mm\_introut | 中断 | O | 0 | 中断流到内存映射通道。 |
| axi\_dma\_tstvec | 那 | O | 0 | 调试内部使用的信号。 |
|  | **AXI4-Lite 接口信号** | | | |
| s\_axi\_lite\_\* | S\_AXI\_LITE | 输入/输出 |  | 有关 AXI4 信号，请参见 AXI 参考*指南* （UG1037） [参考文献 2] 的附录 A。 |
|  | **MM2S 内存映射读取接口信号** | | | |
| m\_axi\_mm2s\_\* | M\_AXI\_MM2S | 输入/输出 |  | 有关 AXI4 信号，请参阅 *AAXI 参考指南* （UG1037） [参考文献 2] 的附录 A。 |
|  | **MM2S 主流接口信号** | | | |
| mm2s\_prmry\_reset\_out\_n | M\_AXIS\_MM2S | O | 1 | 主 MM2S 重置。/有效复位。 |
| m\_axis\_mm2s\_\* | M\_AXIS\_MM2S | 输入/输出 |  | 有关 AXI4 信号，请参见 AXI 参考*指南* （UG1037） [参考文献 2] 的附录 A。 |
|  | **MM2S 主控流接口信号** | | | |
| mm2s\_cntrl\_reset\_out\_n | M\_AXIS\_CNTRL | O | 1 | 控制重置。/有效复位。 |
| m\_axis\_mm2s\_cntrl\_\* | M\_AXIS\_CNTRL | 输入/输出 |  | 有关 AXI4 信号，请参见 AXI 参考*指南* （UG1037） [参考文献 2] 的附录 A。 |
|  | **S2MM 内存映射写入接口信号** | | | |

*表 2-3：* **I/O 信号说明 *（续）***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **信号名称** | **接口** | **信号**  **类型** | **初始化**  **地位** | **描述** |
| m\_axi\_s2mm\_\* | M\_AXI\_S2MM | 输入/输出 |  | 有关 AXI4 信号，请参见 AXI 参考*指南* （UG1037） [参考文献 2] 的附录 A。 |
|  | **S2MM 从数据流接口信号** | | | |
| s2mm\_prmry\_reset\_out\_n | S\_AXIS\_S2MM | O | 1 | 主 S2MM 复位。/有效复位。 |
| s\_axis\_s2mm\_\* | S\_AXIS\_S2MM | 我 | 输入/输出 | 有关 AXI4 信号，请参见 AXI 参考*指南* （UG1037） [参考文献 2] 的附录 A。 |
|  | **S2MM 从机状态流接口信号** | | | |
| s2mm\_sts\_reset\_out\_n | S\_AXIS\_STS | O | 1 | AXI 状态流 （STS） 复位输出。/有效复位。 |
| s\_axis\_s2mm\_sts\_\* | S\_AXIS\_STS | 输入/输出 |  | 有关 AXI4 信号，请参见 AXI 参考*指南* （UG1037） [参考文献 2] 的附录 A。 |
|  | **散射收集内存映射读取接口信号** | | | |
| m\_axi\_sg\_\* | M\_AXI\_SG | 输入/输出 |  | 有关 AXI4 信号，请参见 AXI 参考*指南* （UG1037） [参考文献 2] 的附录 A。 |
|  | **分散 收集 内存 映射 写入 接口 信号** | | | |
| m\_axi\_sg\* | M\_AXI\_SG | 输入/输出 |  | 有关 AXI4 信号，请参见 AXI 参考*指南* （UG1037） [参考文献 2] 的附录 A。 |

## 寄存器空间

用于分散/聚集模式的 AXI DMA 内核寄存器空间如表 2-4 所示。直接寄存器模式的 AXI DMA 内核寄存器空间如表 2-5 所示。AXI DMA 寄存器内存映射到不可缓存的内存空间。此内存空间必须在AXI 字（32 位）边界上对齐。

***注：*** AXI4-Lite 写访问寄存器由 32 位 AXI 写入数据 （\*\_wdata） 信号更新，不受 AXI 写入数据选通 （\*\_wstrb） 信号的影响。对于写入，AXI 写入地址有效 （\*\_awvalid） 和 AXI 写入数据有效 （\*\_wvalid） 信号应同时置位。

### 字节顺序

所有寄存器均采用小端格式，如图2-1所示。

X-参考目标 - 图 2-1

MSB LSB

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 地址偏移0x03 | | 地址偏移0x02 | | 地址偏移0x01 | | 地址偏移0x00 |
| 31 byte3 24 | 23 | byte 2 16 | 15 | byte1 8 | 7 | byte0 0 |

X14584

*图 2-1：***32 位小端示例**

### AXI DMA 寄存器地址映射

*表 2-4：***分散/收集模式寄存器地址映射**

|  |  |  |
| --- | --- | --- |
| **地址空间偏移量**（1） | **名字** | **描述** |
| 00h | MM2S\_DMACR | MM2S DMA 控制寄存器 |
| 04h | MM2S\_DMASR | MM2S DMA 状态寄存器 |
| 08h | MM2S\_CURDESC | MM2S 描述符头指针。低 32 位。 |
| 0Ch | MM2S\_CURDESC\_MSB | MM2S 描述符头指针。高 32 位。 |
| 10h | MM2S\_TAILDESC | MM2S描述符尾指针。低 32 位。 |
| 14h | MM2S\_TAILDESC\_MSB | MM2S描述符尾指针。高 32 位。 |
| 2Ch （2） | SG\_CTL | 分散/收集用户和缓存 |
| 30h | S2MM\_DMACR | S2MM DMA 控制寄存器 |
| 34h | S2MM\_DMASR | S2MM DMA 状态寄存器 |
| 38h | S2MM\_CURDESC | S2MM 描述符头指针。低 32 位 |
| 3Ch | S2MM\_CURDESC\_MSB | S2MM描述符头指针。高 32 位。 |
| 40h | S2MM\_TAILDESC | S2MM描述符尾指针。低 32 位。 |
| 44h | S2MM\_TAILDESC\_MSB | S2MM描述符尾指针。高 32 位。 |

**笔记：**

1. 地址空间偏移量相对于C\_BASEADDR分配。
2. 寄存器 2Ch 仅在多通道模式下配置 DMA 时可用。

*表2-5：***直接寄存器模式寄存器地址映射**

|  |  |  |
| --- | --- | --- |
| **地址空间偏移量**（1） | **名字** | **描述** |
| 00h | MM2S\_DMACR | MM2S DMA 控制寄存器 |
| 04h | MM2S\_DMASR | MM2S DMA 状态寄存器 |
| 08h – 14h | 保留 | 不适用 |
| 18h | MM2S\_SA | MM2S 源地址。较低的 32 位地址。 |
| 1Ch | MM2S\_SA\_MSB | MM2S 源地址。地址的上部 32 位。 |
| 28h | MM2S\_LENGTH | MM2S 传输长度（字节） |
| 30h | S2MM\_DMACR | S2MM DMA 控制寄存器 |
| **地址空间偏移量**（1） | **名字** | **描述** |
| 34h | S2MM\_DMASR | S2MM DMA 状态寄存器 |
| 38h – 44h | 保留 | 不适用 |
| 48h | S2MM\_DA | S2MM 目标地址。较低的 32 位地址。 |
| 4h | S2MM\_DA\_MSB | S2MM 目标地址。上部 32 位地址。 |
| 58h | S2MM\_LENGTH | S2MM 缓冲区长度（字节） |

**笔记：**

1. 地址空间偏移量相对于C\_BASEADDR分配。

### 内存映射到流寄存器详细信息

#### 注册访问类型说明

* RO = 只读。写作没有效果
* R/W = 可读写
* R/WC = 读/写清除

#### MM2S\_DMACR（MM2S DMA 控制寄存器 – 偏移 00h）

此寄存器提供对内存映射到流 DMA 通道的控制。

X-参考目标 - 图 2-2

Dly\_IrqEN

IRQDelay

23

24

31

ERR\_IrqEN

IRQThreshold

回复

重置

RS

钥匙孔

回复

1615141312

4

3

2

1

0

回复

5

6

7

8

9

10

11

循环BD使能

IOC\_IrqEN X14563

*图 2-2* **：MM2S DMACR 寄存器**

*表 2-6：* **MM2S\_DMACR 寄存器详细信息**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **位** | **字段名称** | **默认值** | **访问类型** | **描述** |
| 0 | RS | 0 | R/W | **运行/停止控制，用于控制DMA通道的运行和停止**。   * 0 = 停止 – 当前（如果有）DMA 操作完成后，DMA 停止。对于分散/聚集模式，挂起的命令/传输被刷新或完成。AXI4 流输出可能会提前终止。允许更新队列中的描述符在 引擎停止之前完成对远程内存的更新。   对于直接注册模式，挂起的命令/传输将刷新或完成。AXI4 流输出可能会终止。  当 DMA 引擎停止时，DMA 状态寄存器中的停止位将断言为 1。发生错误时，AXI DMA 硬件会清除此位。CPU 还可以选择清除此位以停止 DMA 操作。   * 1 = 运行 – 启动 DMA 操作。当 DMA 引擎开始操作时，DMA 状态寄存器中的停止位将取消断言为 0。 |
| 1 | 保留 | 1 | RO | 写入此位不起作用，并且始终读取为 1。 |
| 2 | Reset | 0 | R/W | 用于重置 AXI DMA 内核的软复位。将此位设置为 1 会导致重置 AXI DMA。重置将正常完成。  挂起的命令/传输将刷新或完成。  AXI4 流输出可能会提前终止。设置任一 MM2S\_DMACR。重置 = 1 或 S2MM\_DMACR。重置 = 1 重置整个 AXI DMA 引擎。软复位完成后，所有寄存器和位都处于复位状态。   * 0 = 正常操作。 * 1 = 正在重置。 |
| 3 | Keyhole | 0 | R/W | 钥匙孔读取。将此位设置为 1 会导致 AXI DMA 启动  MM2S 读取 （AXI4read） 在非递增地址模式下（固定  AXI4 上的地址突发传输）。当 AXI DMA 处于空闲状态时，可以更新此位。使用锁孔操作时，最大突发长度不应超过 16。启用 DRE 时不应设置此位。  当启用多通道 feature 或处于直接寄存器模式时，此位不起作用。 |
| 4 | Cyclic BD  Enable | 0 | R/W | 设置为 1 时，DMA 在循环缓冲区描述符 （BD） 模式下运行，无需任何用户干预。在此模式下，分散收集模块忽略BD的“已完成”位。设置此位后，您可以循环使用相同的 BD，而不必担心任何过时的描述符错误。  仅当 DMA 空闲或未运行时，才应设置/取消设置此位。在 DMA 运行时更新此位可能会导致意外行为。  当 DMA 在多通道模式下运行时，此位不起作用。 |

*表 2-6：* **MM2S\_DMACR 寄存器问题**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **位** | **字段名称** | **默认值** | **访问类型** | **描述** |
| 11 到 5 | 保留 | 0 | RO | 写入这些位不起作用，它们始终被读取为零。 |
| 12 | IOC\_IrqEn | 0 | R/W | 完成时中断 （IOC） 中断启用。设置为 1 时，允许 DMASR。IOC\_Irq为设置了 IOC 位的描述符生成中断。   * 0 = 禁用 IOC 中断 * 1 = 启用 IOC 中断 |
| 13 | Dly\_IrqEn | 0 | R/W | 中断时延迟计时器中断启用。设置为 1 时，允许 DMASR。Dly\_Irq生成中断。   * 0 = 禁用延迟中断 * 1 = 启用延迟中断   ***注意： 将*** AXI DMA 配置为直接寄存器模式时，将忽略此位。 |
| 14 | Err\_IrqEn | 0 | R/W | 出错时中断启用中断。   * 0 = 错误中断已禁用 * 1 = 启用错误中断 |
| 15 | 保留 | 0 | RO | 写入此位不起作用，它始终读取为零。 |
| 23 到 16 | IRQThreshold | 01h | R/W | 中断阈值。此值用于设置中断阈值。发生 IOC 中断事件时，内部计数器将从中断阈值设置开始倒计时。当计数达到零时，DMA 引擎会生成中断输出。  ***注意：*** 阈值的最小设置为 0x01。写入此寄存器0x00无效。  ***注： 将*** AXI DMA 配置为直接寄存器模式时，将忽略此字段。 |
| 31 到 24 | IRQDelay | 00h | R/W | 中断延迟超时。此值用于设置中断超时值。中断超时机制会导致 DMA 引擎在延迟时间段到期后生成中断。计时器在数据包末尾开始计数，并在收到新数据包或发生超时事件时进行计数。1 超时间隔 = 125 ×（SG 时钟的时钟周期）  在此处设置值 3 会导致延迟超时为 125 x 3 x（SG 时钟的时钟周期）。  ***注：*** 将此值设置为零将禁用 延迟计时器中断。  ***注： 将*** AXI DMA 配置为直接寄存器模式时，将忽略此字段。 |

#### MM2S\_DMASR（MM2S DMA 状态寄存器 – 偏移 04h）

此寄存器提供内存映射到流 DMA 通道的状态。

X-参考目标 - 图 2-3

X14574

23

24

31

IRQThresholdSts

Err\_Irq

IOC\_Irq

Dly\_Irq

SGIncld

DMAIntErr

回复

0

5

161514131211 10

9 8 7 6

4

3

2

1

回复

回复

SGSlvErr

SGDecErr

SGIntErr

回复

DMASlvErr

怠

DMADecErr

停止

IRQDlySts

*图 2-3* **：MM2S DMASR 寄存器**

*表 2-7：* **MM2S\_DMASR 寄存器详细信息**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **位** | **字段名称** | **默认值** | **访问类型** | **描述** |
| 0 | Halted | 1 | RO | DMA 通道已停止。指示 DMA 通道的运行/停止状态。   * 0 = DMA 通道正在运行。 * 1 = DMA 通道已停止。对于分散/聚集模式，此位在DMACR时设置。RS = 0 并且 DMA 和分散收集 （SG） 操作已停止。对于直接寄存器模式 （C\_INCLUDE\_SG = 0），此位在 DMACR 时设置。RS = 0 且 DMA 操作已停止。这可能是DMACR之间的时间滞后。RS = 0 和当 DMASR 时。已停止 = 1。   ***注意：*** 当停止（RS= 0 和停止 = 1）时，在分散收集模式下写入TAILDESC\_PTR指针寄存器对 DMA 操作没有影响。对于直接寄存器模式，写入 LENGTH 寄存器对 DMA 操作没有影响。 |
| 1 | Idle | 0 | RO | DMA 通道空闲。指示 AXI DMA 操作的状态。对于散点/聚集模式，当 IDLE 指示 SG 引擎已到达关联通道的尾指针并且所有排队的描述符都已处理时。写入尾部指针 register 会自动重新启动 DMA 操作。空闲位与 BD 相关联。DMA 可能处于空闲状态，AXI 接口上可能有活动数据。  对于直接寄存器模式，当 IDLE 表示当前传输已完成时。   * 0 = 不空闲。对于分散/聚集模式，SG 尚未到达尾部描述符指针和/或正在进行的 DMA 操作。对于直接注册模式，传输未完成。 * 1 = 空闲。对于分散/聚集模式，SG 已达到尾部描述符指针和 DMA 操作 paused。 对于直接寄存器模式，DMA 传输已完成，控制器已暂停。   ***注意：*** 当通道停止时，此位为 0（DMASR.停止=1）。当 AXI DMA 配置为直接寄存器模式时，在初始传输之前，此位也是 0。 |
| 2 | Reserved | 0 | RO | 写入此位不起作用，它始终读取为零。 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **位** | **字段名称** | **默认值** | **访问类型** | **描述** |
| 3 | SGIncld | C\_  INCLUDE\_  SG | RO | 1 = 已启用分散聚集  0 = 未启用分散聚集 |
| 4 | DMAIntErr | 0 | RO | DMA 内部错误。如果在提取的描述符中指定的缓冲区长度设置为 0，则会发生内部错误。此错误情况会导致 AXI DMA 正常停止。这  DMACR。RS 位设置为 0，当引擎完全关闭时，DMASR。停止位设置为 1。   * 0 = 无 DMA 内部错误 * 1 = 检测到 DMA 内部错误。DMA 引擎停止。   ***注意： 当 AXI DMA*** 配置为直接寄存器模式时，此位不被使用，固定为 0。 |
| 5 | DMASlvErr | 0 | RO | DMA 从站错误。如果从属从存储器映射接口读取发出从属错误，则会发生此错误。此错误情况会导致 AXI DMA 正常停止。DMACR表示。RS 位设置为 0，当引擎完全关闭时，DMASR。Halted 位设置为 1。   * 0 = 无 DMA 从站错误。 * 1 = 检测到 DMA 从站错误。DMA 引擎停止。 |
| 6 | DMADecErr | 0 | RO | DMA 解码错误。如果地址请求指向无效地址，则会发生此错误。此错误情况会导致 AXI DMA 正常停止。DMACR表示。RS 位设置为 0，当引擎完全关闭时，DMASR。停止位设置为 1。   * 0 = 无 DMA 解码错误。 * 1 = 检测到 DMA 解码错误。DMA 引擎停止。 |
| 7 | 保留 | 0 | RO | 写入此位不起作用，并且始终读取为零。 |
| 8 | SGIntErr | 0 | RO | 分散收集内部错误。如果获取已设置“完整位”的描述符，则会发生此错误。有关详细信息，请参阅散点收集描述符部分。这向 SG 引擎指示描述符是过时的描述符。 此错误情况会导致 AXI DMA 正常停止。DMACR表示。RS 位设置为 0，当引擎完全关闭时，DMASR。停止位设置为 1。   * 0 = 无 SG 内部错误。 * 1 = 检测到 SG 内部错误。DMA 引擎停止。   ***注意：*** 当 AXI DMA 配置为直接寄存器模式时，此位不被使用，固定为 0。 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **位** | **字段名称** | **默认值** | **访问类型** | **描述** |
| 9 | SGSlvErr | 0 | RO | 分散收集从站错误。如果在内存映射接口上读取的从属设备发出从属错误，则会发生此错误。此错误情况会导致 AXI DMA 正常停止。DMACR表示。RS 位设置为 0，当引擎完全关闭时，将 DMASR。停止位设置为 1。   * 0 = 无 SG 从站错误。 * 1 = 检测到 SG 从站错误。DMA 引擎停止。   ***注意：*** 当 AXI DMA 配置为直接寄存器模式时，此位不被使用，固定为 0。 |
| 10 | SGDecErr | 0 | RO | 散射收集解码错误。如果CURDESC\_PTR和/或NXTDESC\_PTR指向无效地址，则会发生此错误。此错误情况会导致 AXI DMA 正常停止。这  DMACR。RS 位设置为 0，当引擎完全关闭时，DMASR.Halted 位设置为 1。   * 0 = 无 SG 解码错误。 * 1 = 检测到 SG 解码错误。DMA 引擎停止。   ***注意：*** 当 AXI DMA 配置为直接寄存器模式时，此位不被使用，固定为 0。 |
| 11 | 保留 | 0 | RO | 写入此位不起作用，并且始终读取为零。 |
| 12 | IOC\_Irq | 0 | R/WC | 完成时中断。当分散/聚集模式设置为 1 时，表示描述符完成时生成中断事件。对于设置了帧结束 （EOF） 位的描述符，会发生这种情况。当直接寄存器模式设置为 1 时，表示传输完成时生成了 terrupt 事件。如果在MM2S\_DMACR中启用了相应的位 （IOC\_IrqEn = 1），并且已达到中断阈值，则会导致从 AXI DMA 生成中断输出。   * 0 = 无 IOC 中断。 * 1 = 检测到 IOC 中断。   向此位写入 1 可清除它。 |
| 13 | Dly\_Irq | 0 | R/WC | 延迟中断。设置为 1 时，表示在延迟计时器超时时生成中断事件。如果在MM2S\_DMACR中启用了相应的位 （Dly\_IrqEn = 1），则会从 AXI DMA 生成中断输出。   * 0 = 无延迟中断。 * 1 = 检测到延迟。1 = 检测到 IOC 中断。   向此位写入 1 可清除它。  ***注意：*** 当 AXI DMA 配置为直接寄存器模式时，此位不被使用，固定为 0。 |
| **位** | **字段名称** | **默认值** | **访问类型** | **描述** |
| 14 | Err\_Irq | 0 | R/WC | * 出错时中断。设置为 1 时，表示出错时生成中断事件。如果在MM2S\_DMACR中启用了相应的位 （Err\_IrqEn = 1），则会从 AXI DMA 生成中断输出。向此位写入 1 可清除它。 * 0 = 无错误中断。 * 1 = 检测到错误中断。 |
| 15 | 保留 | 0 | RO | 始终读为零。 |
| 23 至  16 | IRQThresholdSts | 01h | RO | 中断阈值状态。指示当前中断阈值。在 IRQThreshold 字段中编程的值（以 MM2S\_CR 为单位）在每次数据包传输时都会递减并反映在此处。在DMA启动之前或发送第一个数据包之前，寄存器将具有与MM2S\_CR的IRQThreshold字段中编程的值相同的值。  ***注意：*** 仅在启用分散聚集时适用。 |
| 31 至  24 | IRQDelaySts | 00h | RO | 中断延迟时间状态。指示当前中断延迟时间值。  ***注意：*** 仅在启用分散聚集时适用。 |

#### MM2S\_CURDESC（MM2S DMA 流描述符指针寄存器 - 偏移 08h）

此寄存器为内存映射提供当前描述符指针，以流式传输 DMA 散点收集描述符管理。

X-参考目标 - 图 2-4

6

5 4 3 2 1

0

31

当前描述符指针 [31：6] RSVD

X14573

*图 2-4* **：MM2S CURDESC 寄存器**

*表 2-8：* **MM2S\_CURDESC 寄存器详细信息**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **位** | **字段名称** | **默认值** | **访问类型** | **描述** |
| 5 : 0  （偏移量0x38） | 保留 | 0 | RO | 写入这些位不起作用，它们始终被读取为零。 |
| 31 到 6 | 当前描述符指针 | 零 | R/W(RO) | 指示正在处理的当前描述符的指针。在写入TAILDESC\_PTR寄存器之前，此寄存器必须包含指向有效描述符的指针。否则，将出现未定义的结果。当DMACR时。RS 为 1，CURDESC\_PTR变为只读 （RO），用于获取第一个描述符。  当 DMA 引擎运行时 （DMACR.RS=1），  CURDESC\_PTR寄存器由 AXI DMA 更新，以指示当前正在使用的描述符。  在错误检测时，CURDESC\_PTR会更新以反映与检测到的错误关联的描述。  ***注意：*** 寄存器只能在 DMA 引擎停止 （DMACR.RS=0 和 DMASR。已停止 =1）。在所有其他时间，此寄存器是只读 （RO）。描述符必须与 16 个单词对齐，即0x00、0x40、0x80等。任何其他对齐nt 都有未定义的结果。 |

#### MM2S\_CURDES\_MSB（MM2S DMA 流描述符指针寄存器 - 偏移 0Ch）

此寄存器为内存映射提供当前描述符指针的上 32 位，用于流 DMA 散点收集描述符管理。这仅适用于地址空间超过 32 位的情况。

X-参考目标 - 图 2-5

0

31

当前描述符指针 [31：0] X14572

*图 2-5* **：MM2S CURDESC\_MSB寄存器**

*表 2-9：* **MM2S\_CURDESC\_MSB 寄存器详细信息**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **位** | **字段名称** | **默认值** | **访问类型** | **描述** |
| 31 : 0 | 当前  描述符  指针 | 零 | R/W  (RO) | 指示正在处理的当前描述符的指针。在写入TAILDESC\_PTR寄存器之前，此寄存器必须包含指向有效描述符的指针。否则，将出现未定义的结果。当DMACR时。RS 为 1，CURDESC\_PTR变为只读 （RO），用于获取第一个描述符。  当 DMA 引擎运行时 （DMACR.RS=1），  CURDESC\_PTR寄存器由 AXI DMA 更新，以指示当前正在使用的描述符。  在错误检测时，CURDESC\_PTR会更新以反映与检测到的错误关联的描述。  ***注意：*** 寄存器只能在 DMA 引擎停止 （DMACR.RS=0 和 DMASR。已停止 =1）。在所有其他时间，此寄存器是只读 （RO）。描述符必须与 16 个单词对齐，即 0x00、0x40、0x80等。任何其他对齐都有未定义的结果。 |

#### MM2S\_TAILDESC（MM2S DMA 尾部描述符指针寄存器 - 偏移 10h）

此寄存器为内存映射提供尾部描述符指针，以流式传输 DMA 散点收集描述符管理。

X-参考目标 - 图 2-6

31

0

1

2

3

4

5

6

尾部描述符指针 [31：6] RSVD

X14583

*图 2-6：* **MM2S\_TAILDESC 注册**

*表 2-10：* **MM2S\_TAILDESC 寄存器详细信息**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **位** | **字段名称** | **默认值** | **访问类型** | **描述** |
| 5 : 0 | 保留 | 0 | RO | 写入这些位不起作用，它们始终被读取为零。 |
| 31 到 6 | 尾部描述符指针 | 零 | R/W | 指示描述符链中的暂停指针。AXI DMA SG 引擎在完成对当前描述符指针与尾描述符指针匹配的描述符的操作后暂停描述符提取。  当 AXI DMA 通道未停止时（DMASR.停止 = 0），CPU 对TAILDESC\_PTR寄存器的写入会导致 AXI DMA SG 引擎开始获取描述符或在空闲时重新启动 （DMASR.空闲 = 1）。如果不闲着，写  TAILDESC\_PTR除了重新定位暂停 po int 之外没有任何效果。  ***注：*** 软件不得将尾指针移动到尚未更新的位置。软件处理并重新分配所有已完成的描述符 （Cmplted = 1），清除已完成的位，然后移动尾部指针。软件必须将指针指向它更新的最后一个描述符。描述符必须对齐 16 个单词，即 0x00、0x40、0x80等。任何其他对齐都有未定义的结果。 |

#### MM2S\_TAILDESC\_MSB（MM2S DMA 尾部描述符指针寄存器 – 偏移 14h）

此寄存器为内存映射提供上 32 位尾描述符指针，用于流式传输 DMA 散点收集描述符管理。这仅适用于地址速度超过 32 位宽的情况。

X-参考目标 - 图 2-7

31

0

尾部描述符指针 [31：0]

X14582

*图 2-7：* **MM2S\_TAILDESC\_MSB 注册**

*表 2-11：* **MM2S\_TAILDESC\_MSB 寄存器详细信息**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **位** | **字段名称** | **默认值** | **访问类型** | **描述** |
| 31 : 0 | 尾部描述符指针 | 零 | R/W | 指示描述符链中的暂停指针。AXI DMA SG 引擎在完成对当前描述符指针与尾描述符指针匹配的描述符的操作后暂停描述符提取。  当 AXI DMA 通道未停止时 （DMASR.停止 = 0），CPU 对TAILDESC\_PTR\_MSB寄存器的写入会导致 AXI DMA SG 引擎开始获取描述符或在空闲时重新启动 （DMASR.空闲 = 1）。如果它不是空闲的，则写入TAILDESC\_PTR除了重新定位暂停点外，没有效果。  如果 AXI DMA 通道停止（DMASR.已停止 = 1 和 DMACR。RS = 0），CPU 对TAILDESC\_PTR寄存器的写入除了重新定位暂停点外没有任何效果。  ***注：*** 软件不得将指针移动到尚未更新的位置。软件处理并重新分配所有已完成的描述符 （Cmplted = 1），清除已完成的位，然后移动尾部指针。软件必须将指针移动到它更新的最后一个描述符。描述符必须对齐 16 个单词，即 0x00、0x40、0x80等。任何其他对齐都有未定义的结果。 |

#### MM2S\_SA（MM2S DMA 源地址寄存器 – 偏移 18h）

此寄存器提供源地址，用于读取内存映射到流 DMA 传输的系统内存。

X-参考目标 - 图 2-8

31

0

MM2S\_SA

X14579

*图 2-8：* **MM2S\_SA 寄存器**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **位** | **字段名称** | **默认值** | **访问类型** | **描述** |
| 31 : 0 | 源地址 | 零 | R/W | 指示 AXI DMA 读取的源地址的 MSB 32 位，用于将数据传输到 MM2S 通道上的 AXI4 流。  ***注：*** 如果包含数据重新对齐引擎，则源地址可以位于任何字节偏移量。如果未包括数据重新对齐引擎，则 源地址必须与 MM2S 内存映射数据宽度对齐。 |

*表 2-12：* **MM2S\_SA 寄存器详细信息**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **位** | **字段名称** | **默认值** | **访问类型** | **描述** |
| 31 ： 0 | 源地址 | 零 | R/W | 指示 AXI DMA 读取的源地址，用于将数据传输到 MM2S 通道上的 AXI4-Stream。  ***注：*** 如果包含数据重新对齐引擎，则源地址可以位于任何字节偏移量。如果未包括数据重新对齐引擎，则源地址 必须与 MM2S 内存映射数据宽度对齐。 |

#### MM2S\_SA\_MSB（MM2S DMA 源地址寄存器 – 偏移 1Ch）

此寄存器提供源地址的高 32 位，用于读取系统内存，以便内存映射到流 DMA 传输。 仅当 DMA 配置为大于 32 的地址空间时，这才适用。

X-参考目标 - 图 2-9

31

0

MM2S\_SA\_MSB

X14578

*图 2-9：* **MM2S\_SA\_MSB 注册**

*表 2-13：* **MM2S\_SA\_MSB 寄存器详细信息**

#### MM2S\_LENGTH（MM2S DMA 传输长度寄存器 — 偏移 28h）

此寄存器提供从系统内存读取并传输到 MM2S AXI4 流的字节数。

X-参考目标 - 图 2-10

保留

0

31

2625

MM2S 长度 [25：0] 1

注1：由C\_SG\_Length\_Width确定的有效寄存器位

X14575

*图 2-10：* **MM2S\_LENGTH 寄存器**

*表 2-14：* **MM2S\_LENGTH 寄存器详细信息**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **位** | **字段名称** | **默认值** | **访问类型** | **描述** |
| 25（1） 到 0 | 长度 | 零 | R/W | 指示要为 MM2S 通道传输的字节数。将非零值写入此寄存器将启动 MM2S 传输。 |
| 31 到 26 | 保留 | 0 | RO | 写入这些位不起作用，它们始终被读取为零。 |

**笔记：**

1. 长度宽度字段由缓冲区长度寄存器宽度参数确定。最小宽度为 8 位（7 到 0），最大宽度为 26 位（25 到 0）。

***SG\_CTL（分散/收集用户和缓存控制寄存器 - 偏移 2Ch） 仅当*** DMA 配置为多通道模式时，此寄存器才可用。

X-参考目标 - 图 2-11

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 11 | SG\_USER | 8 | 7 | 回复 | 4 | 3 | SG\_CACHE 0 |

31 保留 12

*图 2-11：* **SG\_CTL 寄存器**

*表 2-15：* **SG\_CTL 寄存器详细信息**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **位** | **字段名称** | **默认值** | **访问类型** | **描述** |
| 3 到 0 | SG\_CACHE | 0011b | R/W | 分散/收集缓存控制。写入此寄存器中的值反映M\_AXI\_SG接口的m\_axi\_sg\_arcache和m\_axi\_sg\_awcache信号。 |
| 7 到 4 | 保留 | 0 | RO | 写入这些位不起作用，它们始终被读取为零。 |
| 11 到 8 | SG\_USER | 0 | R/W | 分散/收集用户控件。写入此寄存器中的值反映M\_AXI\_SG接口的m\_axi\_sg\_aruser和m\_axi\_sg\_awuser信号。 |

### 流到内存映射寄存器详细信息

***S2MM\_DMACR （S2MM DMA 控制寄存器 – 偏移 30h）***

此寄存器提供对流到内存映射 DMA 通道的控制。

X-参考目标 - 图 2-12

Dly\_IrqEN

IRQDelay

23

24

31

ERR\_IrqEN

IRQThreshold

回复

重置

RS

钥匙孔

回复

1615141312

4

3

2

1

0

回复

5

6

7

8

9

10

11

循环BD使能

IOC\_IrqEN x14594

*图 2-12：* **S2MM DMACR 寄存器**

*表 2-16：* **S2MM\_DMACR 寄存器详细信息**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **位** | **字段名称** | **默认值** | **访问类型** | **描述** |
| 0 | RS | 0 | R/W | 运行/停止控制，用于控制 DMA 通道的运行和停止。   * 0 = 停止 – 当前（如果有）DMA 操作完成后，DMA 停止。对于分散/聚集模式，挂起的命令/传输被刷新或完成。AXI4 流可能会提前终止。允许更新队列中的描述符在引擎停止之前完成对远程内存的更新。对于直接寄存器模式，挂起的命令/传输将刷新或完成。AXI4 流可能已终止。无法保证 S2MM AXI4 上的数据完整性。   当 DMA 引擎停止时，DMA 状态寄存器中的停止位将断言为 1。发生错误时，AXI DMA 硬件会清除此位。CPU 还可以选择清除此位以停止 DMA 操作。   * 1 = 运行 – 启动 DMA 操作。 当 DMA 引擎开始操作时，DMA 状态寄存器中的停止位将取消断言为 0。 |
| 1 | 保留 | 1 | RO | 写入此位不起作用，并且始终读取为 1。 |
| 2 | 重置 | 0 | R/W | 用于重置 AXI DMA 内核的软复位。将此位设置为 1 会导致重置 AXI DMA。重置将正常完成。  挂起的命令/传输将刷新或完成。  AXI4 流输出提前终止，如有必要，与相关的  啧。设置任一MM2S\_DMACR。重置 = 1 或  S2MM\_DMACR。重置 = 1 重置整个 AXI DMA 引擎。软复位完成后，所有寄存器和位都处于复位状态。   * 0 = 重置未进行中。正常操作。 * 1 = 正在重置。 |

*表 2-16：* **S2MM\_DMACR 寄存器详细信息 *（续）***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **位** | **字段名称** | **默认值** | **访问类型** | **描述** |
| 3 | 钥匙孔 | 0 | R/W | 锁孔写入。将此位设置为 1 会导致 AXI DMA 启动  S2MM 写入（AXI4 写入）以非递增地址模式（AXI4 上的固定地址突发传输）。当 AXI DMA 处于空闲状态时，可以修改此位。启用钥匙孔操作时，最大突发长度不能超过 16。启用 DRE 时，此位不应为 se t。  在多通道模式下使用 DMA 时，此位不起作用。 |
| 4 | 循环BD使能 | 0 | R/W | 设置为 1 时，可以在循环缓冲区描述符 （BD） 模式下使用 DMA，而无需任何用户干预。在此模式下，分散收集模块忽略BD的“已完成”位。使用此功能，您可以循环使用相同的BD，而不必担心任何过时的描述符错误。  当 DMA 在多通道模式或直接寄存器模式下运行时，此位不起作用。 |
| 11 到 5 | 保留 | 0 | RO | 写入这些位不起作用，它们始终被读取为零。 |
| 12 | IOC\_IRqEn | 0 | R/W | 完全中断时中断启用。设置为 1 时，允许“完成时中断”事件为设置了“完成位”位的描述符生成中断。   * 0 = 禁用 IOC 中断。 * 1 = 已启用 IOC 中断。 |
| 13 | Dly\_IrqEn | 0 | R/W | 中断时延迟计时器中断启用。设置为 1 时，允许错误事件生成中断输出。   * 0 = 禁用延迟中断。 * 1 = 启用延迟中断。   ***注意：*** 仅在启用分散聚集时适用。 |
| 14 | Err\_IrqEn | 0 | R/W | 出错时中断启用中断。设置为 1 时，允许错误事件生成中断输出。   * 0 = 错误中断已禁用。 * 1 = 启用错误中断。 |
| 15 | 保留 | 0 | RO | 写入此位不起作用，并且始终读取为零。 |

*表 2-16：* **S2MM\_DMACR 寄存器详细信息 *（续）***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **位** | **字段名称** | **默认值** | **访问类型** | **描述** |
| 23 到 16 | IRQThreshold | 01h | R/W | 中断阈值。此值用于设置中断阈值。发生 IOC 中断事件时，内部计数器将从中断阈值设置开始倒计时。当计数达到零时，DMA 引擎会生成中断输出。  ***注意：*** 阈值的最小设置为 0x01。写入此寄存器0x00无效。  ***注意：*** 仅在启用分散聚集时适用。 |
| 31 到 24 | IRQDelay | 00h | R/W | 中断延迟超时。此值用于设置中断超时值。中断超时是一种机制，用于在延迟时间段到期后使 DMA 引擎生成中断。计时器在数据包末尾开始计数 ，并在收到新数据包或发生超时事件时重置。  1 超时间隔 = 125 ×（SG 时钟的时钟周期）  在此处设置值 3 会导致延迟超时为 125 x 3 x（SG 时钟的时钟周期）。  ***注：*** 将此值设置为零将禁用延迟计时器中断。  ***注意：*** 仅在启用分散聚集时适用。 |

#### S2MM\_DMASR（S2MM DMA 状态寄存器 – 偏移 34h）

此寄存器提供流到内存映射 DMA 通道的状态。

X-参考目标 - 图 2-13

x14595

23

24

31

IRQThresholdSts

Err\_Irq

IOC\_Irq

Dly\_Irq

SGIncld

DMAIntErr

回复

0

5

161514131211 10

9 8 7 6

4

3

2

1

回复

回复

SGSlvErr

SGDecErr

SGIntErr

回复

DMASlvErr

怠

DMADecErr

停止

IRQDlySts

*图 2-13：* **S2MM DMASR 寄存器**

**S2MM\_DMASR 注册详细信息**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **位** | **字段名称** | **默认值** | **访问类型** | **Description** |
| 0 | 停止 | 1 | RO | DMA 通道已停止。指示 DMA 通道的运行/停止状态。   * 0 = DMA 通道正在运行。 * 1 = DMA 通道已停止。对于分散/聚集模式，此位在 DMACR 时设置。RS = 0，DMA 和 SG 操作已停止。对于直接寄存器模式，此位在 DMACR 时设置。RS = 0 且 DMA 操作已停止。DMACR之间可能存在时间滞后 。RS = 0 和当 DMASR 时。已停止 = 1。   ***注意：***当停止（RS= 0 和停止 = 1）时，在分散收集模式下写入TAILDESC\_PTR指针寄存器对 DMA 操作没有影响。对于直接寄存器模式，写入 LENGTH 寄存器对 DMA 操作没有影响。 |
| 1 | 怠 | 0 | RO | DMA 通道空闲。指示 AXI DMA 操作的状态。对于散点/聚集模式，当 IDLE 指示 SG 引擎已到达关联通道的尾指针并且所有排队的描述符都已处理时。写入尾部指针 register 会自动重新启动 DMA 操作。  对于直接寄存器模式，当 IDLE 表示当前传输已完成时。   * 0 = 不空闲。 * 1 = 空闲。   ***注意：*** 当通道停止时，此位为 0（DMASR.停止=1）。当 AXI DMA 配置为直接寄存器模式时，在初始传输之前，此位也是 0。 |
| 2 | 保留 | 0 | RO | 写入此位不起作用，它始终读取为零。 |
| 3 | SGIncld | C\_  INCLUDE\_  SG | RO | 包括分散收集引擎。DMASR。SGIncld = 1 表示包含分散收集引擎，并且 AXI DMA 配置为分散收集模式。DMASR。SGIncld = 0 表示已排除分散收集引擎，并且 AXI DMA 已配置为直接寄存器模式。 |
| 4 | DMAIntErr | 0 | RO | DMA 内部错误。如果在提取的描述符中指定的缓冲区长度设置为 0，则会发生此错误。此外，在分散收集模式下并使用状态应用程序长度字段时，当状态 AXI4 流数据包 RxLength 字段与S\_AXIS\_S2MM接口接收的 S2MM 数据包不匹配时，会发生此错误。禁用分散收集后，如果在内存写入期间发生任何错误，或者传入数据包大于 DMA 长度寄存器中指定的数据包，则会标记此错误。  此错误情况会导致 AXI DMA 正常停止。DMACR表示。RS 位设置为 0，当引擎完全关闭时，DMASR。停止位设置为 1。   * 0 = 无 DMA 内部错误。 * 1 = 检测到 DMA 内部错误。 |

**S2MM\_DMASR 注册 Detaills *（续）***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **位** | **字段名称** | **默认值** | **访问类型** | **描述** |
| 5 | DMASlvErr | 0 | RO | DMA 从站错误。如果从属从存储器映射接口读取发出从属错误，则会发生此错误。此错误情况会导致 AXI DMA 正常停止。这  DMACR。RS 位设置为 0，当引擎完全关闭 DMASR 时。停止 位设置为 1。   * 0 = 无 DMA 从站错误。 * 1 = 检测到 DMA 从站错误。 |
| 6 | DMADecErr | 0 | RO | DMA 解码错误。如果地址请求指向无效地址，则会发生此错误。此错误情况会导致 AXI DMA 正常停止。DMACR表示。RS 位设置为 0，并且  当发动机完全关闭时，DMASR。停止位设置为 1。   * 0 = 无 DMA 解码错误。 * 1 = 检测到 DMA 解码错误。 |
| 7 | 保留 | 0 | RO | 写入此位不起作用，它始终读取为零。 |
| 8 | SGIntErr | 0 | RO | 分散收集内部错误。如果获取已设置完整位的描述符，则会发生此错误。这向 SG 引擎指示描述符是尾部描述符。此错误情况会导致 AXI DMA 正常停止。这  DMACR。RS 位设置为 0，当引擎完全关闭时，DMASR。停止位设置为 1。   * 0 = 无 SG 内部错误。 * 1 = 检测到 SG 内部错误。   此错误无法记录到描述符中。  ***注意：仅当***启用分散聚集时适用 |
| 9 | SGSlvErr | 0 | RO | 分散收集从站错误。如果在内存映射接口上读取的从属设备发出从属错误，则会发生此错误。此错误情况会导致 AXI DMA 正常停止。DMACR表示。RS 位设置为 0，当引擎完全关闭时，将 DMASR。停止位设置为 1。   * 0 = 无 SG 从站错误。 * 1 = 检测到 SG 从站错误。DMA 引擎停止。   此错误无法记录到描述符中。  ***注意：*** 仅在启用分散聚集时适用。 |
| 10 | SGDecErr | 0 | RO | 散射收集解码错误。如果CURDESC\_PTR和/或NXTDESC\_PTR指向无效地址，则会发生此错误。此错误情况会导致 AXI DMA 正常停止。这  DMACR。RS 位设置为 0，当引擎完全关闭时，DMASR。停止位设置为 1。   * 0 = 无 SG 解码错误。 * 1 = 检测到 SG 解码错误。DMA 引擎停止。   此错误无法记录到描述符中。  ***注意：*** 仅在启用分散聚集时适用。 |

**S2MM\_DMASR 注册详细信息 *（续）***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **位** | **字段名称** | **默认值** | **访问类型** | **描述** |
| 11 | 保留 | 0 | RO | 写入此位不起作用，它始终读取为零。 |
| 12 | IOC\_Irq | 0 | R/WC | 完成时中断。当分散/聚集模式设置为 1 时，表示描述符完成时生成中断事件。对于设置了帧结束 （EOF） 位的描述符，会发生这种情况。如果直接寄存器模式设置为 1，则表示传输完成时生成了 inte rrupt 事件。  如果启用了S2MM\_DMACR中的相应位 （IOC\_IrqEn = 1），并且已达到中断阈值，则会导致从 AXI DMA 生成中断输出。   * 0 = 无 IOC 中断。 * 1 = 检测到 IOC 中断。   向此位写入 1 可清除它。 |
| 13 | Dly\_Irq | 0 | R/WC | 延迟中断。设置为 1 时，表示在延迟计时器超时时生成中断事件。如果启用了S2MM\_DMACR中的相应位 （Dly\_IrqEn = 1），则会从 AXI DMA 生成中断输出。   * 0 = 无延迟中断。 * 1 = 检测到延迟间中断。   向此位写入 1 可清除它。  ***注意：*** 仅在启用分散聚集时适用。 |
| 14 | Err\_Irq | 0 | R/WC | 出错时中断。设置为 1 时，表示出错时生成中断事件。如果相应的位在  启用S2MM\_DMACR （Err\_IrqEn = 1），将从 AXI DMA 生成中断输出。向此位写入 1 可清除它。   * 0 = 无错误中断。 * 1 = 检测到错误中断。 |
| 15 | 保留 | 0 | RO | 写入此位不起作用，它始终读取为零。 |
| 23 到 16 | IRQThresholdSts | 01h | RO | 中断阈值状态。指示当前中断阈值。在 IRQThreshold 字段中编程的值（以 S2MM\_CR 为单位）在每次 pa cket 传输时都会递减并反映在此处。在DMA启动之前或接收第一个数据包之前，此寄存器将具有与IRQThreshold字段中编程的相同值 S2MM\_CR。  ***注意：*** 仅在启用分散聚集时适用。 |
| 31 到 24 | IRQDelaySts | 00h | RO | 中断延迟时间状态。指示当前中断延迟时间值。  ***注意：*** 仅在启用分散聚集时适用。 |

#### S2MM\_CURDESC（S2MM DMA 电流描述符指针寄存器 – 偏移 38h）

此r egister 为流到内存映射 DMA 分散收集描述符管理提供当前描述符指针。

X-参考目标 - 图 2-14

31

0

1

2

3

4

5

6

当前描述符指针 [31：6] RSVD

x14592

*图 2-14：* **S2MM 寄存器**

*表 2-18：* **S2MM\_CURDESC 寄存器详细信息**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **位** | **字段名称** | **默认值** | **访问类型** | **描述** |
| 5 : 0  （偏移量  0x38） | 保留 | 0 | RO | 写入这些位不起作用，它们始终被读取为零。 |
| 31 到 6 | 当前  描述符  指针 | 零 | R/W  (RO) | 指示当前正在处理的缓冲区描述符的指针。在写入TAILDESC\_PTR寄存器之前，此寄存器必须包含指向有效描述符的指针。否则，将出现未定义的结果。当DMACR时。RS 为 1，CURDESC\_PTR变为仅 Re ad （RO），用于获取第一个描述符。  当 DMA 引擎运行时 （DMACR.RS=1），  CURDESC\_PTR寄存器由 AXI DMA 更新，以指示当前正在使用的描述符。  在检测到错误时，CURDESC\_PTR会更新以反映与检测到的错误关联的描述符。  ***注意：*** 寄存器只能在 DMA 引擎停止时由 CPU 写入 （DMACR.RS=0 和 DMASR。已停止 =1）。在所有其他时间，此寄存器是只读 （RO）。  缓冲区描述符必须是 16 个单词的aliged，即 0x00、0x40、0x80 等。任何其他对齐都有未定义的结果。 |

#### S2MM\_CURDESC\_MSB（S2MM DMA 电流描述符指针寄存器 – 偏移 3Ch）

此寄存器为流到内存映射 DMA 散点收集描述符管理提供当前描述符指针的前 32 位。仅当为大于 32 位的地址空间配置 DMA 时，才使用此方法。

X-参考目标 - 图 2-15

31

0

当前描述符指针 [31：0]

x14591

*图 2-15：* **S2MM CURDESC\_MSB寄存器**

*表 2-19：* **S2MM\_CURDESC\_MSB 寄存器详细信息**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **位** | **字段名称** | **默认值** | **访问类型** | **描述** |
| 31 : 0 | 当前  描述符  指针 | 零 | R/W  (RO) | 指示当前正在处理的缓冲区描述符的指针。在写入TAILDESC\_PTR寄存器之前，此寄存器必须包含指向有效描述符的指针。否则，将出现未定义的结果。当DMACR时。RS 为 1，CURDESC\_PTR变为 只读 （RO），用于获取第一个描述符。  当 DMA 引擎运行时 （DMACR.RS=1），  CURDESC\_PTR寄存器由 AXI DMA 更新，以指示当前正在使用的描述符。  检测到错误时，CURDESC\_PTR会更新以反映 与检测到的错误关联的描述符。  ***注意：*** 寄存器只能在 DMA 引擎停止时由 CPU 写入 （DMACR.RS=0 和 DMASR。已停止 =1）。在所有其他时间，此寄存器是只读 （RO）。 |

#### S2MM\_TAILDESC（S2MM DMA 尾部描述符指针寄存器 – 偏移 40h）

此寄存器为流到内存映射 DMA 散点收集描述符管理提供尾描述符指针。

X-参考目标 - 图 2-16

31

0

1

2

3

4

5

6

尾部描述符指针 [31：6] RSVD

x14600

*图 2-16：* **S2MM 尾部寄存器**

*表 2-20：* **S2MM\_TAILDESC 寄存器详细信息**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **位** | **字段名称** | **默认值** | **访问类型** | **描述** |
| 5 : 0 | 保留 | 0 | RO | 写入这些位不起作用，它们始终被读取为零。 |
| 31 到 6 | 尾部描述符指针 | 零 | R/W | 指示描述符链中的暂停指针。AXI DMA SG 引擎在完成对当前描述符指针与尾描述符指针匹配的描述符的操作后暂停描述符提取。  当 AXI DMA 通道未停止时（DMASR.停止 = 0），CPU 写入TAILDESC\_PTR寄存器会导致 AXI DMA SG 引擎开始获取描述符或在空闲时重新启动  （DMASR.空闲 = 1）。如果它不是空闲的，则写入TAILDESC\_PTR除了重新定位暂停点外没有任何效果。  如果 AXI DMA 通道 DMACR。RS 位设置为 0 （DMASR.已停止 = 1 和 DMACR。RS = 0），CPU 对TAILDESC\_PTR寄存器的写入除了 重新定位暂停点外没有任何效果。  ***注：*** 软件不得将尾部指针移动到尚未更新的位置。软件处理并重新分配所有已完成的描述符（Cmplted = 1），清除已完成的位，然后移动尾部指针。软件必须将指针移动到它更新的最后一个描述符。  描述符必须对齐 16 个单词，即 0x00、0x40、0x80等。任何其他对齐都有未定义的结果。 |

#### S2MM\_TAILDESC\_MSB（S2MM DMA 尾部描述符指针寄存器 – 偏移 44h）

此寄存器为流到内存映射 DMA 散点收集描述符管理提供上 32 位尾描述符指针。当 DMA 配置为大于 32 的广告着装空间时，将使用此选项。

X-参考目标 - 图 2-17

31

0

尾部描述符指针 [31：0]

x14599

*图 2-17：* **S2MM TAILDESC\_MSB寄存器**

*表 2-21：* **S2MM\_TAILDESC\_MSB 寄存器详细信息**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **位** | **字段名称** | **默认值** | **访问类型** | **描述** |
| 31 : 0 | 尾部描述符指针 | 零 | R/W | 指示描述符链中的暂停指针。AXI DMA SG 引擎在完成对当前描述符指针与尾描述符指针匹配的描述符的操作后暂停描述符提取。  当 AXI DMA 通道未停止时（DMASR.停止 = 0），CPU 对TAILDESC\_PTR\_MSB寄存器的写入会导致 AXI DMA SG 引擎开始获取描述符或在空闲时重新启动 （DMASR.空闲 = 1）。如果它不是空闲的，那么写TAILDESC\_PTR除了重新定位暂停点之外没有任何效果。  如果 AXI DMA 通道 DMACR。RS 位设置为 0 （DMASR.已停止 = 1 和 DMACR。RS = 0），CPU 对TAILDESC\_PTR寄存器的写入除了重新定位暂停点外没有任何效果。  ***注：*** 软件不得将尾部指针移动到未更新的位置。软件处理并重新分配所有已完成的描述符 （Cmplted = 1），清除已完成的位，然后移动尾部指针。软件必须将指针移动到它更新的最后一个描述符。 |

#### S2MM\_DA（S2MM DMA 目标离子地址寄存器 – 偏移 48h）

此寄存器提供目标地址，用于写入系统内存，以便流到内存映射到 DMA 传输。

X-参考目标 - 图 2-18

31

0

S2MM\_DA

x14593

*图 2-18：* **S2MM\_DA 寄存器**

*表 2-22：* **S2MM\_DA 寄存器详细信息**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **位** | **字段名称** | **默认值** | **访问类型** | **描述** |
| 31 : 0 | 目的地地址 | 零 | R/W | 指示 AXI DMA 写入的目标地址，用于从 S2MM 通道上的 AXI4 流传输数据。  ***注：*** 如果包含数据重新对齐引擎，则目标地址可以位于任何字节偏移量。如果未包含数据重新对齐引擎，则目标地址必须与 S2MM 内存映射数据宽度对齐。 |

***S2MM\_DA\_MSB（S2MM DMA 目标地址寄存器 – 偏移 4Ch）***

此寄存器提供目标地址的上 32 位，用于写入系统内存以进行流到内存映射到 DMA 的传输。仅当为大于 32 的地址空间配置 DMA 时，才使用此方法。

X-参考目标 - 图 2-19

31

0

S2MM\_DA

x14593

*图 2-19：* **S2MM\_DA\_MSB 寄存器**

*表 2-23：* **S2MM\_DA\_MSB 寄存器详细信息**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **位** | **字段名称** | **默认值** | **访问类型** | **描述** |
| 31 : 0 | 目的地地址 | 零 | R/W | 指示 AXI DMA 写入的目标地址的 MSB 32 位，以便从 S2MM 通道上的 AXI4 流传输数据。  ***注：*** 如果包含数据重新对齐引擎，则目标地址可以位于任何字节偏移量。如果未包括数据重新对齐引擎，则目标地址必须与 S2MM 内存映射数据宽度对齐。 |

#### S2MM\_LENGTH （S2MM DMA 缓冲器长度寄存器 – 偏移 58h）

此寄存器提供缓冲区的长度（以字节为单位），用于将数据从流到内存映射 DMA 传输写入。

X-参考目标 - 图 2-20

0

31

2625

保留

注 1：有效寄存器位由 C\_SG\_Length\_Width S2MM 长度 [25：0] 1 确定

X14569

*图 2-20：* **S2MM\_LENGTH 寄存器**

*表 2-24：* **S2MM\_LENGTH 寄存器详细信息**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **位** | **字段名称** | **默认值** | **访问类型** | **描述** |
| 25（1） 到 0 | 长度 | 零 | R/W | 指示可用于从 S2MM 通道写入接收数据的 S2MM 缓冲区的长度（以字节为单位）。将非零值写入此寄存器可使 S2MM 通道接收数据包数据。  S2MM 传输完成后，写入 S2MM AXI4 接口上的实际字节数将更新到S2MM\_LENGTH寄存器。  ***注意：*** 此值必须大于或等于 S2MM AXI4 流上要接收的最大预期数据包。 小于接收数据包的值会导致未定义的行为。 |
| 31 到 26 | 保留 | 0 | RO | 写入这些位不起作用，它们始终被读取为零。 |

**笔记：**

1. 长度宽度字段由缓冲区长度寄存器宽度参数确定。最小宽度为 8 位（7 到 0），最大宽度为 26 位（25 到 0）。

## 分散收集描述符

本节定义 S2MM（接收）和 MM2S（传输）散射收集描述符的字段，用于将 AXI DMA 配置为散射/聚集模式。描述符由 8 个 32 位基本字和 0 或 5 个用户应用程序字组成。d描述符将来支持 64 位地址并支持用户应用程序数据。通过帧开始和帧结束标志支持每个数据包的多个描述符。还包括已完成状态和完成时中断。Buffer 长度可以描述每个描述符多达 67、108、863 字节的数据缓冲区。两个数据传输方向 MM2S 和 S2MM 需要两个描述符链。

*表 2-25：* **描述符字段（非多通道模式）**

|  |  |  |
| --- | --- | --- |
| **地址空间偏移**量（1） | **名字** | **描述翻录** |
| 00h | NXTDESC | 下一个描述符指针 |
| 04h | NXTDESC\_MSB | 下一个描述符指针的高 32 位 |
| 08h | BUFFER\_ADDRESS | 缓冲区地址 |
| 0Ch | BUFFER\_ADDRESS\_MSB | 缓冲区地址的高 32 位。 |
| 10h | RESERVED | 不适用 |
| 14h | RESERVED | 不适用 |
| 18h | CONTROL | 控制 |
| 1Ch | STATUS | 地位 |
| 20h | APP0 | 用户应用领域 0 （2） |
| 24h | APP1 | 用户应用字段 1 |
| 28h | APP2 | 用户应用字段 2 |
| 2Ch | APP3 | 用户应用字段 3 |
| 30h | APP4 | 用户应用领域 4 |

**笔记：**

1. 地址空间偏移相对于系统内存中的 16 - 32 位字对齐方式，即 0x00、0x40、0x80 等。
2. 用户应用程序字段（APP0、APP1、APP2、APP3 和 APP4）仅在包含控制/状态流时使用，当不包括控制/状态流时，分散收集引擎不会获取或更新用户应用程序字段。
3. 仅当为大于 32 的地址空间配置 DMA 时，才使用 MSB 字段或 32 位以上的地址。

### MM2S\_NXTDESC（MM2S 下一个描述符指针）

此值提供指向描述符链中下一个描述符的指针。

X-参考目标 - 图 2-21

6

5 4 3 2 1

0

31

下一个描述符指针 [31：6] 保留

X14577

*图 2-21：* **MM2S\_NXTDESC**

*表 2-26：* **MM2S\_NXTDESC 详细信息**

|  |  |  |
| --- | --- | --- |
| **位** | **字段名称** | **描述** |
| 5 : 0 | 保留 | 这些位是保留的，应设置为零。 |
| 31 : 6 | 下一个描述符指针 | 指示指向下一个描述符的第一个单词的低阶指针。  ***注意：*** 描述符必须与 16 个单词对齐，即 0x00、 0x40、 0x80等。任何其他对齐都有未定义的结果。 |

### MM2S\_NXTDESC\_MSB（MM2S 下一个描述符指针）

此值提供指向描述符链中下一个描述符的指针的前 32 位。仅当为大于 32 的地址空间配置 AXI DMA 时，才使用此选项。

X-参考目标 - 图 2-22

31

0

下一个描述符指针 [31：0]

X14576

*图 2-22：* **MM2S\_NXTDESC\_MSB**

*表 2-27：* **MM2S\_NXTDESC\_MSB 详细信息**

|  |  |  |
| --- | --- | --- |
| **位** | **字段名称** | **描述** |
| 31 : 0 | 下一个描述符指针 | 指示指向下一个描述符的第一个单词的指针的 MSB 32 位。 |

### MM2S\_BUFFER\_ADDRESS（MM2S 缓冲区地址）

此值提供指向要从系统内存传输到流的数据缓冲区的指针。

X-参考目标 - 图 2-23

0

31

缓冲区地址 [31：0] X14570

*图 2-23：* **MM2S 缓冲区地址**

*表 2-28：* **MM2S\_BUFFER\_ADDRESS 详细信息**

|  |  |  |
| --- | --- | --- |
| **位** | **字段名称** | **描述** |
| 31 : 0 | 缓冲区地址 | 提供要从内存映射传输到流的数据的位置。  ***注：*** 如果包含数据重新对齐引擎，则缓冲区地址可以位于任何字节偏移量，但缓冲区中的数据必须是连续的。如果数据  不包括重新对齐引擎，缓冲区地址必须与 MM2S 内存映射数据宽度对齐。 |

### MM2S\_BUFFER\_ADDRESS\_MSB（MM2S 缓冲区地址）

此值提供指向要从系统内存传输到流的数据缓冲区的上 32 位指针。仅当为大于 32 的地址空间配置 AXI DMA 时，才使用此选项。

X-参考目标 - 图 2-24

0

31

缓冲区地址 [31：0] X14570

*图 2-24：* **MM2S 缓冲区地址**

*表 2-29：* **MM2S\_BUFFER\_ADDRESS\_MSB 详细信息**

|  |  |  |
| --- | --- | --- |
| **位** | **字段名称** | **描述** |
| 31 : 0 | 缓冲区地址 | 提供要从内存映射传输到流的数据位置的 MSB 32 位。 |

### MM2S\_CONTROL（MM2S 控制）

此值提供对从内存映射到流的 MM2S 传输的控制。

X-参考目标 - 图 2-25

0

缓冲区长度 [25：0]

回复

TXSOF

31

30

2928

27

26

25

TXEOF X14571-061019

*图 2-25：* **MM2S\_CONTROL**

*表 2-30：* **MM2S\_CONTROL 详细信息**

|  |  |  |
| --- | --- | --- |
| **位** | **字段名称** | **描述** |
| 25 : 0 | 缓冲长度 | 指示传输缓冲区的大小（以字节为单位）。此值指示要在 MM2S 流上传输出去的字节数。缓冲区长度的可用宽度由**参数缓冲区长度寄存器宽度**指定。此字段最多可以描述 67，108，863 字节的传输。在微模式下配置AXI\_DMA时，此值不应超过以下公式：  （MM2S 内存映射数据宽度/8）\*Burst\_length  ***注意：*** 将缓冲器长度寄存器宽度设置为小于26会降低FPGA资源利用率。 |
| 26 | 传输结束  框架(TXEOF) | 帧结束。指示要处理的最后一个缓冲区的标志。此标志由 CPU 设置，以向 AXI DMA 指示此描述符描述数据包的结尾。与此描述符关联的缓冲区最后传输。   * 0 = 不是帧结束。 * 1 = 帧结束。   ***注意：*** 为了正确操作，每个数据包必须有一个帧开始 （SOF） 描述符 （TXSOF=1） 和一个帧结束 （EOF） 描述符 （TXEOF=1）。让单个描述符描述整个数据包是有效的，该数据包是机器人h TXSOF=1 和 TXEOF=1 的描述符。 |
| 27 | TXSOF | 帧的开始。指示要处理的第一个缓冲区的标志。此标志由 CPU 设置，以向 AXI DMA 指示此描述符描述数据包的开始。首先传输与此描述符关联的缓冲区。   * 0 = 不是 frame 的开始。 * 1 = 帧的开始。   ***注意：*** 启用状态控制流后，从帧开始 （SOF） 描述符 （TXSOF=1） 的 APP0 到 APP4 的用户应用程序数据将在控制流输出上传输。 |
| 31 到 28 | 保留 | 此位是保留的，应写入为零。 |

### MM2S\_STATUS（MM2S 状态）

此值提供从内存映射到流的 MM2S 传输的状态。

X-参考目标 - 图 2-26

0

传输的字节数 [25：0]

回复

DMAIntErr

DMASlvErr

31

30

29

28

27

26

25

X14581

Cmplt

DMADecErr

*图 2-26：* **MM2S\_STATUS**

*表 2-31：* **MM2S\_STATUS 详细信息**

|  |  |  |
| --- | --- | --- |
| **位** | **字段名称** | **描述** |
| 25 : 0 | 传输的字节数 | 指示为此描述符传输的实际数据的大小（以字节为单位）。此值指示要在 MM2S 流上传输出去的字节数。此值应与“控制缓冲区长度”字段匹配。  传输字节的可用宽度由**参数缓冲区长度寄存器的宽度**指定。此字段最多可以描述 67，108，863 字节的传输。在微模式下配置时，AXI\_DMA不会更新这些字段。  ***注意：*** 将 缓冲器长度寄存器宽度设置为小于23会降低FPGA资源利用率。在微模式下配置AXI\_DMA时，此字段不会更新。 |
| 27 to 26 | 保留 | 这些位是保留的，应设置为零。 |
| 28 | DMAIntErr | DMA 内部错误。主 AXI 数据移动器检测到内部错误。如果要传输的 0 长度字节被馈送到 AXI 数据移动器，则会发生此错误。仅当提取的描述符中指定的缓冲区长度设置为 0 时，才会发生这种情况。  此错误导致AXI DMA 正常停止。DMACR表示。RS 位设置为 0，当引擎完全关闭时，DMASR。停止位设置为 1。   * 0 = 无 DMA 内部错误。 * 1 = 检测到 DMA 内部错误。DMA 引擎停止。 |
| 29 | DMASlvErr | DMA 从站错误。主 AXI 数据移动器检测到从错误。如果从属从存储器映射接口读取发出从属错误，则会发生此错误。此错误情况会导致 AXI DMA 正常停止。DMACR表示。RS 位设置为 0，当 engine 完全关闭时，DMASR。停止位设置为 1。   * 0 = 无 DMA 从站错误。 * 1 = 检测到 DMA 从站错误。DMA 引擎停止。 |

*表 2-31：* **MM2S\_STATUS 详细信息 *（续）***

|  |  |  |
| --- | --- | --- |
| **位** | **字段名称** | **描述** |
| 30 | DMADecErr | DMA 解码错误。解码 主 AXI 数据移动器检测到的错误。如果描述符缓冲区地址指向无效地址，则会发生此错误。此错误情况会导致 AXI DMA 正常停止。DMACR表示。RS 位设置为 0，当引擎 完全关闭时，DMASR。停止位设置为 1。   * 0 = 无 DMA 解码错误。 * 1 = 检测到 DMA 解码错误。DMA 引擎停止。 |
| 31 | Cmplt | 完成。这向软件指示 DMA 引擎已完成关联描述符所述的传输。传输完成后，DMA 引擎将此位设置为 1。该软件可能会操作任何将 Complet 位设置为 1 的描述符。   * 0 = 描述符未完成。 * 1 = 描述符已完成。   ***注意：*** 如果在将此位设置为 1 的情况下获取描述符，则该描述符被视为过时的描述符。标记了 SGIntErr，AXI DMA 引擎停止。 |

### MM2S\_APP0到MM2S\_APP4（MM2S 用户应用程序字段 0 到 4）

此值为 MM2S 控制流提供用户应用程序字段。

X-参考目标 - 图 2-27

31

0

APPn

X14580

*图 2-27：* **MM2S\_STATUS**

*表 2-32：* **用户应用程序详细信息**

|  |  |  |
| --- | --- | --- |
| **位** | **字段名称** | **描述** |
| 31 : 0 | APP0 到 APP4 | 用户应用程序字段 0 到 4。指定特定于用户的应用程序数据。什么时候  启用状态控制流，帧开始 （SOF） 描述符的应用程序 （APP） 字段将传输到 AXI 控制流。对于 SOF = 0 的其他 MM2S 描述符，将提取 APP 字段，但会忽略这些字段。  ***注：*** 未启用状态控制流时，不会对这些字段进行列分。 |

### S2MM\_NXTDESC（S2MM 下一个描述符指针）

此值提供指向描述符链中下一个描述符的指针。

X-参考目标 - 图 2-28

31

0

1

2

3

4

5

6

下一个描述符指针 [31：6] 保留

x14597

*图 2-28：* **S2MM\_NXTDESC**

*表 2-33：* **S2MM\_NXTDESC 详细信息**

|  |  |  |
| --- | --- | --- |
| **位** | **字段名称** | **描述** |
| 5 : 0 | 保留 | 这些位是保留的，应设置为零。 |
| 31 到 6 | 下一个描述符指针 | 指示指向下一个描述符的第一个单词的低阶指针。  ***注意：*** 描述符必须与 16 个单词对齐，即0x00、0x40、0x80等。任何其他对齐都有未定义的结果。 |

### S2MM\_NXTDESC\_MSB（S2MM 下一个描述符指针）

此值提供指向 des criptor 链中下一个描述符的指针的上 32 位。仅当为大于 32 的地址空间配置 AXI DMA 时，才使用此选项。

X-参考目标 - 图 2-29

31

0

下一个描述符指针 [31：0]

x14596

*图 2-29：* **S2MM\_NXTDESC\_MSB**

*表 2-34：* **S2MM\_NXTDESC\_MSB 详细信息**

|  |  |  |
| --- | --- | --- |
| **位** | **字段名称** | **描述** |
| 31 : 0 | 下一个描述符指针 | 指示指向下一个描述符的第一个单词的指针的 MSB 32 位。 |

### S2MM\_BUFFER\_ADDRESS（S2MM 缓冲区地址）

此值提供指向可用于将数据从流传输到系统内存的缓冲区空间的指针。

X-参考目标 - 图 2-30

31

0

缓冲区地址 [31：0]

x14588

*图 2-30：* **S2MM 缓冲区地址**

*表 2-35：* **S2MM\_BUFFER\_ADDRESS 详细信息**

|  |  |  |
| --- | --- | --- |
| **位** | **字段名称** | **描述** |
| 31 : 0 | 缓冲区地址 | 提供可用于存储从流传输到内存映射的数据的缓冲区空间的位置。  ***注：*** 如果包含数据重新对齐引擎，则缓冲区地址可以位于任何字节偏移量。如果未包括数据重新对齐引擎，则缓冲区地址ss 必须与 S2MM 内存映射数据宽度对齐。 |

### S2MM\_BUFFER\_ADDRESS\_MSB（S2MM 缓冲区地址）

此值提供指针的前 32 位，指向可用于将数据从流传输到系统内存的缓冲区空间。仅当为大于 32 的地址空间配置 AXI DMA 时，才使用此选项。

X-参考目标 - 图 2-31

31

0

缓冲区地址 [31：0]

x14588

*图 2-31：* **S2MM 缓冲区地址 （MSB）**

*表 2-36：* **S2MM\_BUFFER\_ADDRESS\_MSB 详细信息**

|  |  |  |
| --- | --- | --- |
| **位** | **字段名称** | **描述** |
| 31 : 0 | 缓冲区地址 | 提供 MSB 32 位缓冲区空间的位置，可用于存储从流传输到内存映射的数据。 |

### S2MM\_CONTROL（S2MM 控制）

此值提供从流到内存映射的 S2MM 传输的控制。

X-参考目标 - 图 2-32

RXEOF餐厅

0

31302928272625

回复 缓冲区长度 [25：0]

RXSOF x14590

*图 2-32：* **S2MM\_CONTROL**

*表 2-37：* **S2MM\_CONTROL 详细信息**

|  |  |  |
| --- | --- | --- |
| **位** | **字段名称** | **描述** |
| 31 到 28 | 保留 | 这些位是保留的，应设置为零。 |
| 27 | RXSOF | 帧的开始。指示要处理的第一个缓冲区的标志。此标志由软件/用户设置，以向 AXI DMA 指示此描述符描述数据包的开始。  首先接收与此描述符关联的缓冲区。   * 0 = 不是帧的开始 。 * 1 = 帧的开始。   这仅适用于在微模式下配置AXI\_DMA。 |
| 26 | 接收帧结束 | 帧结束。指示要处理的最后一个缓冲区的标志。此标志由 sw/user 设置，以向 AXI DMA 指示此描述符描述数据包的结尾。  最后接收与此描述符关联的缓冲区。   * 0 = 不是帧结束。 * 1 = 帧结束。   这仅适用于在微模式下配置AXI\_DMA。 |
| 25 : 0 | 缓冲长度 | 此值指示可用于接收 S2MM 流中的数据的空间量（以字节为单位）。缓冲区长度的可用宽度由**参数缓冲区长度寄存器宽度指定。** 此字段最多可以描述 67，108，863 字节的传输。  ***注意：*** S2MM 描述符链中的总缓冲区空间（即链中每个描述符的缓冲区长度值的总和）必须至少能够容纳最大接收数据包大小。如果收到大于定义的缓冲区空间的数据包，则会出现未定义的结果。  ***注意：*** 将缓冲器长度寄存器宽度设置为小于23会降低FPGA资源利用率。  ***注意：*** 在微模式下配置AXI\_DMA时，此值不应超过以下公式：  （S2MM 内存映射数据宽度/8）\*Burst\_length |

### S2MM\_STATUS（S2MM 状态）

此值提供从流到内存映射的 S2MM 传输的状态。

X-参考目标 - 图 2-33

传输的字节数 [25：0]

0

x143598

31302928272625

RXEOF餐厅

RXSOF

DMASlvErr

Cmplt

DMAIntErr

DMADecErr

*图 2-33：* **S2MM\_STATUS**

*表 2-38：* **S2MM\_STATUS 详细信息**

|  |  |  |
| --- | --- | --- |
| **位** | **字段名称** | **描述** |
| 25 : 0 | 传输的字节数 | 此值指示在此描述符描述的缓冲区中接收和存储的数据量。这可能与缓冲区长度匹配，也可能不匹配。例如，如果此描述符指示缓冲区长度为 1，024 字节，但仅接收 50 个字节并存储在缓冲区中，则“传输的字节数”字段指示 0x32。通过将 RXSOF 描述符中每个描述符的传输字节值添加到帧接收端 （RXEOF） 描述符，可以确定整个接收数据包长度。  ***注意：***传输字节的可用宽度由参数缓冲区**长度寄存器的宽度**指定。此字段最多可以描述 67，108，863 字节的传输。  ***注意：*** 将缓冲器长度寄存器宽度设置为小于26会降低FPGA资源利用率。在微模式下配置AXI\_DMA时，此字段不会更新。 |
| 26 | RXEOF | 帧结束。指示缓冲区保存数据包的最后一部分的标志。此位由 AXI DMA 设置，以向软件/用户指示与此描述符关联的缓冲区包含数据包的末尾。   * 0 = 不是帧结束。 * 1 = 帧结束。   ***注意：***启用控制/状态流时，通过状态流输入发送的用户应用程序许可数据存储在 RXEOF 描述符的 APP0 到 APP4 中。 |
| 27 | RXSOF | 帧的开始。指示缓冲区保存数据包第一部分的标志。此位由 AXI DMA 设置，以向 sw/user 指示与此描述符关联的缓冲区包含数据包的开头。   * 0 = 不是帧的开始。 * 1 = 帧的开始。 |
| 28 | DMAIntErr | DMA 内部错误。主 AXI 数据移动器检测到内部错误。如果要传输的 0 长度字节被馈送到 AXI 数据移动器，则会发生此错误。仅当提取的描述符中指定的缓冲区长度设置为 0 时，才会发生这种情况。如果运行不足或超限情况，也可能导致此 e rror。  此错误情况会导致 AXI DMA 正常停止。DMACR表示。RS 位设置为 0，当引擎完全关闭时，DMASR。停止位设置为 1。   * 0 = 无 DMA 内部错误或。 * 1 = 检测到 DMA 内部错误。DMA 引擎停止。 |

*表 2-38：* **S2MM\_STATUS 详细信息 *（续）***

|  |  |  |
| --- | --- | --- |
| **位** | **字段名称** | **描述** |
| 29 | DMASlvErr | DMA 从站错误。主 AXI 数据移动器检测到从错误。如果从属从存储器映射接口读取发出从属错误，则会发生此错误。此错误情况会导致 AXI DMA 正常停止。DMACR表示。RS 位设置为 0，当 engine 完全关闭时，DMASR。停止位设置为 1。   * 0 = 无 DMA 从站错误。 * 1 = 检测到 DMA 从站错误。DMA 引擎停止。 |
| 30 | DMADecErr | DMA 解码错误。解码 主 AXI 数据移动器检测到的错误。如果描述符缓冲区地址指向无效地址，则会发生此错误。此错误情况会导致 AXI DMA 正常停止。DMACR表示。RS 位设置为 0，当引擎 完全关闭时，DMASR。停止位设置为 1。   * 0 = 无 DMA 解码错误。 * 1 = 检测到 DMA 解码错误。DMA 引擎停止。 |
| 31 | Cmplt | 完成。这向软件指示 DMA 引擎已完成关联描述符所述的传输。传输完成后，DMA 引擎将此位设置为 1。该软件可以操作任何描述符，将 Completed 位设置为 1。   * 0 = 描述符未完成。 * 1 = 描述符已完成。   ***注意：*** 如果在将此位设置为 1 的情况下获取描述符，则该描述符被视为过时的描述符。标记了 SGIntErr，AXI DMA 引擎停止。 |

### S2MM\_APP0 到 S2MM\_APP3（S2MM 用户应用领域 0 到 3）

此值为状态流上的 S2MM 已接收状态提供用户应用程序字段空间。

X-参考目标 - 图 2-34

31

0

APPn

x14586

*图 2-34：* **S2MM\_APP0 至 S2MM\_APP3**

*表 2-39：* **用户应用程序 0 到 3 详细信息**

|  |  |  |
| --- | --- | --- |
| **位** | **字段名称** | **描述** |
| 31 : 0 | APP0 到 APP3 | 启用状态/控制流后，AXI 状态上接收的状态数据  流存储在帧结束 （EOF） 描述符的 APP 字段中。对于 EOF = 0 的其他 S2MM 描述符，APP 字段由散点收集 Engi ne 设置为零。  ***注：*** 如果禁用了状态/控制字段，则分散收集引擎不会更新这些字段。 |

### S2MM\_APP4（S2MM用户应用领域4）

此值为状态流上的 S2MM 接收状态提供用户应用程序 4 字段空间。

X-参考目标 - 图 2-35

0

22

接收长度

31

23

APP4 x14587

*图 2-35：* **S2MM\_APP4**

*表 2-40：* **用户应用程序 4 详细信息**

|  |  |  |
| --- | --- | --- |
| **位** | **字段名称** | **描述** |
| 31 : 0 | APP4 / RxLength | 用户应用程序字段 4 和接收字节长度。如果未启用“在状态流中使用 RxLength”，则此字段的功能与 APP0 到 APP3 相同，因为在 AXI 状态流上接收的状态数据存储到帧结束 （EOF） 描述符的 APP4 字段中。  启用**“在状态流中使用 RxLength**”时，此字段具有双重用途。缓冲区长度寄存器宽度中指定的前几个最低有效位指定在 S 2MM 主数据流上接收的数据包的接收字节总数。其次，剩余最重要的位是用户应用程序数据。 |

### 描述符管理

在开始 DMA 操作之前，软件应用程序必须设置描述符链。当 AXI DMA 开始处理描述符时，它会提取、处理并更新描述符。通过分析描述符，软件应用程序读取相关DMA传输的状态，获取接收（S2MM）通道上的用户信息，并确定传输的完成情况。有了这些信息，软件应用程序就可以管理描述符和数据缓冲区。

软件应用程序处理与已完成描述符关联的每个缓冲区，并重新分配描述符以供 AXI DMA 使用。为了防止软件和硬件相互踩踏，创建了尾指针模式。尾指针由软件初始化，指向描述符链的末端。这将成为硬件的暂停点。当硬件开始运行时，它会获取并处理链中的每个描述符，直到它到达尾指针。然后，AXI DMA 暂停描述符处理。允许软件 处理和重新分配任何完整位设置为 1 的描述符。

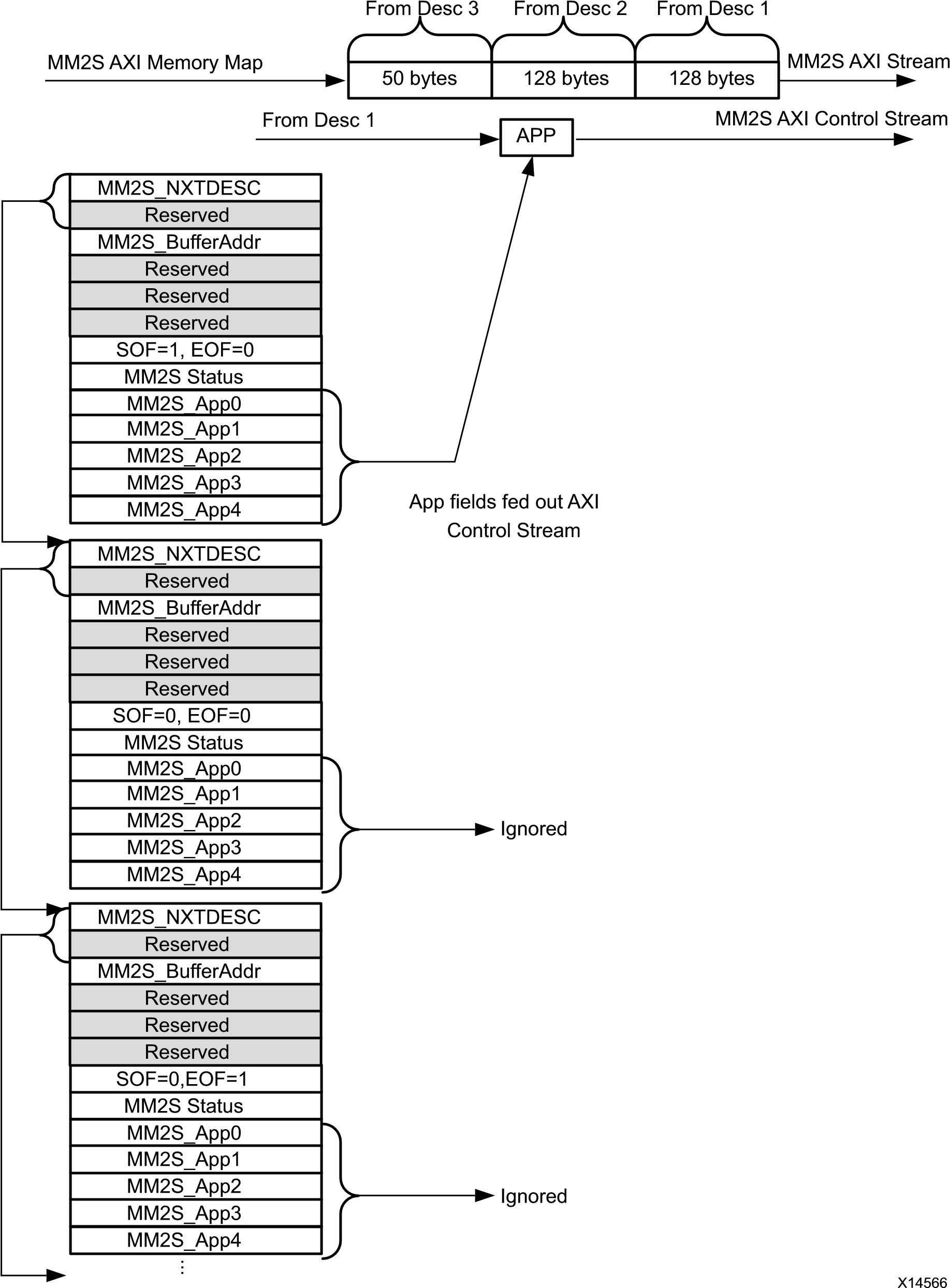
写入 TAILDESC 寄存器的操作会导致 AXI DMA 硬件（如果它在尾部指针处暂停）再次开始处理描述符。如果 AXI DMA 硬件未在 TAILDESC指针处暂停，则写入 TAILDESC 寄存器对硬件没有影响。在这种情况下，AXI DMA 将继续处理描述符，直到到达新的尾部描述符指针位置。描述符管理必须由软件完成。

AXI DMA 不管理描述符。

### MM2S 描述符设置和 AXI 控制流

描述符 SOF/EOF 设置与 AXI 控制流之间的关系如图 2-36 所示。SOF=1 的描述符是数据包的开头，并重置 MM2S 方向的 DRE。 如果启用了状态/控制流，则此描述符的用户应用程序字段也会显示在 AXI 控制流上。AXI DMA 引擎将忽略 SOF=1 描述符后面的用户应用程序字段，包括 EOF =1 的描述符。如果禁用了状态/控制流，则 SG 提取引擎不会获取**用户应用程序**字段。

X-参考目标 - 图 2-36



*图 2-36：* **描述符与 MM2S 流和控制流的关系的详细信息**

### AXI 控制流

AXI 控制流从分散收集描述符提供给目标设备，以获取用户应用程序数据。控制数据与 MM2S 主数据流相关联，可以在主数据包之前、期间或之后从 AXI DMA 发送出去。允许目标设备进行限制，并且可以通过 AXI DMA 进行限制。 图 2-37 显示了描述符用户应用程序 fields 如何在 AXI 控制流上显示的示例。AXI DMA 向目标设备插入一个指示数据类型的标志。这是作为第一个单词发送的。对于以太网，控制标记0xA在第一个单词的四个最高有效位 （MSB） 中。

X-参考目标 - 图 2-37

分散聚集说明

旗

AXI\_DMA附加标志

（A000\_0000）

m\_axis\_mm2s\_cntrl\_aclk

m\_axis\_mm2s\_cntrl\_tdata[31：0]

m\_axis\_mm2s\_cntrl\_tkeep[3：0]

m\_axis\_mm2s\_cntrl\_tvalid

m\_axis\_mm2s\_cntrl\_tready

m\_axis\_mm2s\_cntrl\_tlast

用户应用程序字段被驱动

m\_axis\_mm2s\_cntrl\_tdata

APP0

APP1

APP3

APP4

APP2

Fh

MM2S\_NXTDESC

保留

MM2S 缓冲区地址

保留

保留

保留

MM2S 控制 SOF=1

MM2S 状态

MM2S\_App0

MM2S\_App1

MM2S\_App2

MM2S\_App3

MM2S\_App4

X14568

*图 2-37：* **MM2S 控制流的示例用户应用程序字段/计时**

### S2MM 描述符设置和 AXI 状态流

描述符 RXSOF/RXEOF 设置与 AXI 状态流之间的关系如图 2-38 所示。RXSOF=1 的描述符描述了包含接收数据包第一部分的缓冲区。RXEOF=1 的描述符描述了包含接收数据包最后一部分的 b uffer。

为了正确操作，软件必须指定足够的缓冲区空间（描述符链的每个描述符中的缓冲区长度之和），使其大于或等于接收的最大大小数据包。

X-参考目标 - 图 2-38

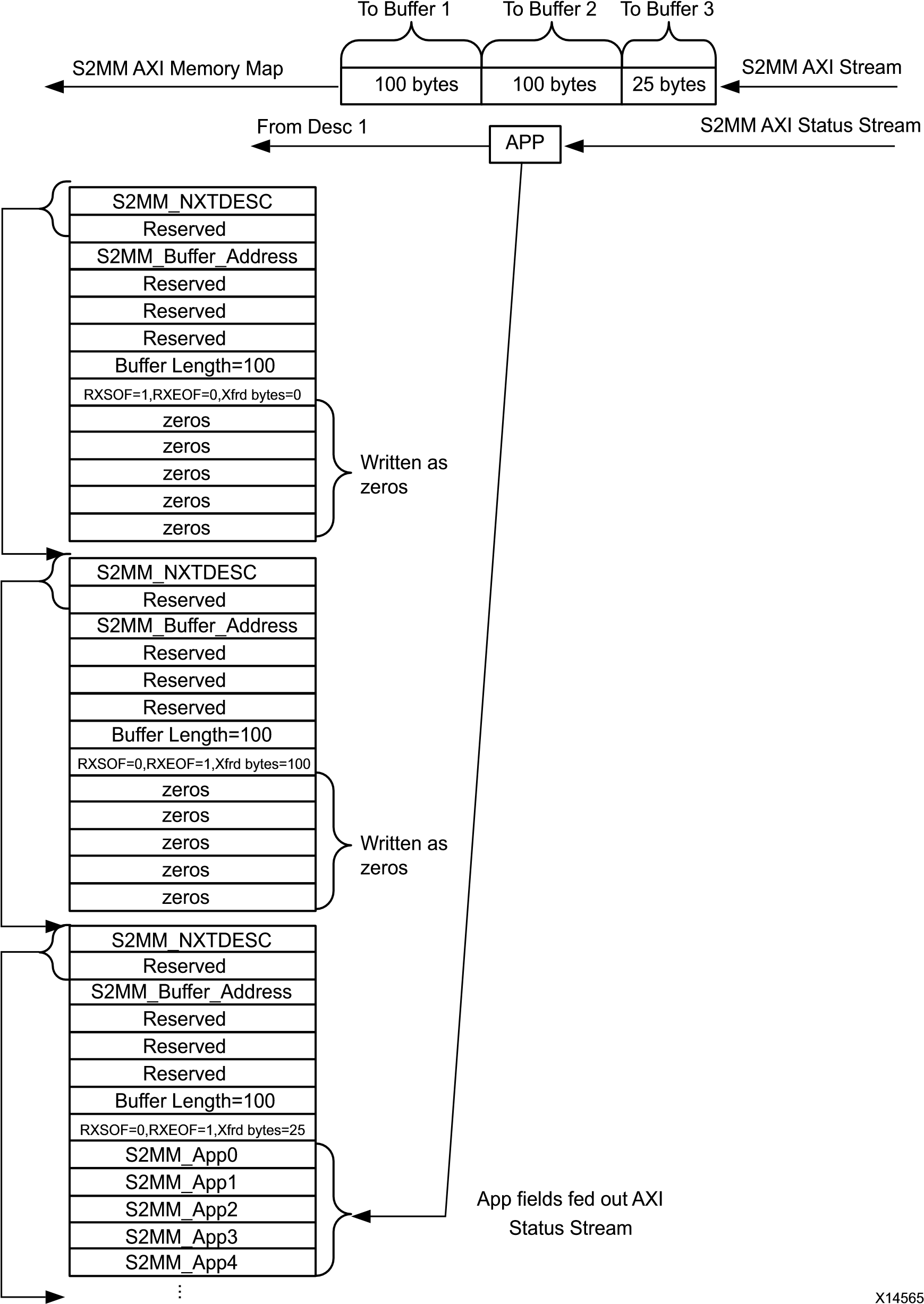


图 2-38： 描述符与 S2MM 流和状态流的关系的详细信息

如果包含状态/控制流，则接收的状态将存储在设置了 RXEOF 的描述符的用户应用程序字段（APP0 到 APP4）中。

特定缓冲区的接收和存储数据的实际字节计数将更新为关联描述符中的“传输的字节数”字段。该软件可以通过将描述符从 RXSOF 遍历到 RXEOF 并添加字节传输d 字段来获得总字节数来确定接收了多少字节。对于在状态流中提供总长度的应用程序，此值存储在 RXEOF=1 的描述符中的用户定义的应用程序位置。

### AXI 状态流

提供 AXI 状态流，用于将目标设备状态传输到分散收集描述符中的用户应用程序数据字段。状态数据与 S2MM 主数据流相关联。如图 2-39 所示，状态数据包更新到描述数据包的检测到的 last 描述符 （RXEOF = 1） 的应用字段。通常，状态流应位于 S2MM 数据流的开头。如果禁用了在状态流中使用 RxLength ，则状态流可以在 S2MM 帧过程中随时出现。 帧结束 （EOF） 缓冲区描述符 （BD） 更新仅在收到整个状态流时发生。

X-参考目标 - 图 2-39

分散聚集说明

旗

追加的目标设备

旗帜 （5xxx\_xxxxh）

s\_axis\_s2mm\_sts\_aclk

s\_axis\_s2mm\_sts\_tdata[31：0]

s\_axis\_s2mm\_sts\_tkeep[3：0]

s\_axis\_s2mm\_sts\_tvalid

s\_axis\_s2mm\_sts\_tready

s\_axis\_s2mm\_sts\_tlast

用户应用程序字段填充

s\_axis\_s2mm\_sts\_tdata

APP0

APP1

APP3

APP4

APP2

Fh

恩特德斯克

保留

缓冲区地址

保留

保留

保留

缓冲长度

Cntrl/STS （EOF =1）

应用0

应用1

应用2

应用3

应用4

X14567

*图 2-39：* **S2MM 状态流的示例用户应用程序字段/计时**

## 多通道 DMA 支持

**重要说明：***AXI DMA 的多渠道支持将停止。有关多通道支持，请参阅 AXI 多通道直接内存访问 （PG288） [参考文献 12]。*

多通道模式使 DMA 能够连接到流端的多个主站和从站。添加了一组与源和目标信令相关的新信号。它们是：

* TID – 5 位信号。用户定义的边带信令。
* TDEST – 5 位信号。为数据流提供粗略的路由信息。
* tuser – 4 位信号。用户定义的边带信令。

### 散点聚集模式 （C\_INCLUDE\_SG = 1）

添加了新的描述符字段以支持多通道和二维传输。正如AXI DMA多通道操作中所述，AXI DMA支持高效的二维内存访问模式，通过AXI4流通道传输二维块。内存访问模式由三个参数控制：HSIZE、VSIZE 和 STRIDE。每个数据包的多重描述符通过数据包开始和数据包结束标志提供支持。

在此模式下，Vivado® 集成设计环境 （IDE） IP 定制功能可禁用状态/控制流。

AXI DMA 可以通过启用多通道模式并在 MM2S 和 S2MM 路径上选择所需的通道数来设置多通道模式 e。

#### MM2S （TX） 描述符

X-参考目标 - 图 2-40

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 31 | | | | | | | NXTDESCPTR | | | | | | | 6 | | 5 | Rsvd 0 0 |
| 31 | | | | | | | NXTDESCPTR\_MSB | | | | | | |  | | | 0 |
| 31 | | | | | | | Buffer Address | | | | | | |  | | | 0 |
| 31 | | | | | | | Buffer Address （MSB） | | | | | | |  | | | 0 |
| 31 ARUSER 28 | | | | 27 AWCACHE | | | 24 | 23 Rsvd 20 | | 19 TUSER16 | | 15 Rsvd 13 | 12 TID | 8 | 7 Rsvd 5 | | 4 TDEST 0 |
| 31 VSIZE | | | | | | | 19 | | 18 Rsvd 16 | | 15 | | | Stride | | | 0 |
| 31 | 30 | 29 | 28 | TX SOP | TX  EOP | 25 | Rsvd 16 | | | | 15 | | | HSIZE | | | 0 |
| Cmp | DE | SE | IE | 27 | | | 保留 | | | | | | |  | | | 0 |

0x00

0x04

0x08

0x0C

0x10

0x14

0x18 0x1C

x12597

*图 2-40：* **TX 描述符**

**TX 描述符字段**

|  |  |  |
| --- | --- | --- |
| **地址空间偏移量** | **名字** | **描述** |
| 00h | NXTDESC | 位 5：0 – 保留  位 31：6 – 下一个描述符指针 |
| 04h | NXTDESC MSB | 提供下一个描述符指针的前 32 位。当 AXI DMA 配置为大于 32 的地址空间时适用。 |
| 08h | BUFFER\_ADDRESS | 位 31：0 – 缓冲区地址  提供要从内存映射传输到流的数据的位置。地址应与内存映射数据宽度对齐。 |
| 0Ch | BUFFER\_ADDRESS | 提供缓冲区地址的高 32 位。仅当为大于 32 的地址空间配置 AXI DMA 时，这才适用。 |
|  |  | 多通道控制位。  位 4：0 – TDEST 为数据流提供路由信息。TDEST 值对于整个数据包是静态的。  TX 描述符字段中提供的 TDEST 值显示在流端的 TDEST 信号上。 |
| • 位 7：5 – 保留 |
| 10h | MC\_CTL | • 位 12：8 – TID：提供流标识符。TID 值对于整个数据包是静态的。TX 描述符字段中提供的 TID 值显示在流端的 TID 信号上。 |
| • 位 15：13 – 保留 |
| • 位 19：16 – TUSER：用于用户定义信息的边带信号。TUSER 值对于整个数据包是静态的。TX 描述符字段中提供的 TUSER 值显示在流端的 TUSER 信号上。 |
| • 位 23：20 – 保留 |
| • 位 27：24 – ARCACHE：缓存类型。此信号提供有关传输的可缓存特征的其他信息。有关不同的解码机制，请参阅 *AMBA® AXI 和 ACE 协议规范* [参考文献 3]。  来自 TX 描述的 ARCACHE 值在地址周期期间显示在 ARCACHE [3：0] 总线上。此字段的默认值为 0011。 |

**TX 描述符字段 *（续）***

|  |  |  |
| --- | --- | --- |
| **地址空间偏移量** | **名字** | **描述** |
|  |  | • 位 31：28 – ARUSER：用于用户定义信息的边带信号。来自 TX 描述符的 ARUSER 值显示在 ARUSER [3：0] 上。ARUSER 值及其解释是用户定义的。您可以通过对链内所有描述符中的相同值进行编程来使整个数据包的 ARUSER 保持静态。 |
| 14h | STRIDE\_VSIZE | * 位 15：0 – 步幅控制。它是连续“水平”读取的第一个地址之间的地址距离。   读取将从缓冲区地址开始并读取 HSIZE 字节，然后跳过 STRIDE-HSIZE 地址并读取 HSIZE 字节，依此类推。这一直持续 到读取 VSIZE 行为止。在 AXI4-Stream 上，它作为连续数据包在m\_axis\_mm2s\_接口上传输出去，并在传输的最后一个数据节拍时以单个断言 TLAST 终止。   * 位 18：16 – 保留 * 位 31：19 – 步幅访问的“水平线”数量。可以表示二维视频数据或二维矩阵的大小。这是每个pa cket 预期传输的传输数，每个 HSIZE 字节长。 |
| 18h | HSIZE | * 位 15：0 – 从连续连续字节地址在每条“水平线”中传输的字节数。当按行主要顺序读取矩阵时，可以表示视频行的一部分或矩阵行的一部分。 * 位 25：16 – 保留 * 位 26 – TXEOP – 数据包结束标志。它指示与此描述符关联的缓冲区最后传输。此标志由 CPU 设置。   1. – 不是数据包结束。1 – 数据包结束 * 位 27 – TXSOP – 数据包标志的开始。它指示 首先传输与此描述符关联的缓冲区。此标志由 CPU 设置。   1. – 不是数据包的开始。1 – 数据包开始 * 位 31：28–保留 |

**TX 描述符字段 *（续）***

|  |  |  |
| --- | --- | --- |
| **地址空间偏移量** | **名字** | **描述** |
| 1Ch | MC\_STS | 多通道状态位。   * 位 27：0 – 保留 * 位 28 – IE – DMA 由于运行不足或超限而导致的内部错误。   1. – 没有 DMA 内部错误   2. – 检测到 DMA 内部错误。DMA 引擎停止。 * 位 29 – SE – DMA 从站错误。如果从属从存储器映射接口读取发出从属错误，则会发生此错误。   1. – 没有 DMA 从站错误   2. – 检测到 DMA 从站错误。DMA 引擎停止 * 位 30 – DE – DMA 解码错误。如果地址要求st 是无效地址，则会发生此错误。   1. – 无 DMA 解码错误   2. – 检测到 DMA 解码错误。DMA 引擎停止 * 位 31 – CMP – 已完成。这向软件指示 DMA 引擎已完成传输。   1. – 描述符未完成   2. – 描述符完成 |

**笔记：**

1. 从AXI读取的角度来看，ARCACHE，ARUSER值很重要。应根据需要在描述符中指定这些值。对于正常操作，ARCACHE 应设置为 0011，而 ARUSER 可以设置为 0000。
2. VSIZE 上的值为 0 是非法的，并且多通道 DMA 中的值无法按预期运行。

#### *表 2-41：*S2MM （RX） 描述符

X-参考目标 - 图 2-41

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 31 | | | |  | | | NXTDESCPTR | | | | | | | 6 |  | 5 | Rsvd 0 0 |
| 31 | | | |  | | | NXTDESCPTR\_MSB | | | | | | |  |  | | 0 |
| 31 | | | |  | | | Buffer Address | | | | | | |  |  | | 0 |
| 31 | | | |  | | | Buffer Address (MSB) | | | | | | |  |  | | 0 |
| 31 AWUSER 28 | | | | 27 AWCACHE | | | 24 | 23 保留 | | | | | |  |  | | 0 |
| 31 | | | | VSIZE | | | 19 | | | 18 Rsvd 16 | | 15 | | Stride |  | | 0 |
| 31 | | | | 保留 | | | 16 | | | | | 15 | | HSIZE |  | | 0 |
| Cmp | DE | SE | IE | RXSOP | RXEOP | 25 | 24 | | 23 Rsvd20 | | 19 TUSER 16 | | 15 Rsvd 13 | 12 TID 8 | 7 Rsvd 5 | | 4 TDEST 0 |

0x00

0x04

0x08

0x0C

0x10

0x14

0x18

0x1C

x12597

*图 2-41：* **RX 描述符**

**RX 描述符字段**

|  |  |  |
| --- | --- | --- |
| **地址空间偏移量** | **名字** | **描述** |
| 00h | NXTDESC | 位 5：0 – 保留  位 31：6 – 下一个描述符指针 |
| 04h | NXTDESC\_MSB | 提供下一个描述符指针的高 32 位。当为大于 32 的地址空间配置 DMA 时适用。 |
| 08h | BUFFER\_ADDRESS | 位 31：0 – 缓冲区地址  提供可用于存储从流传输到内存映射的数据的缓冲区空间的位置。地址应与内存映射数据宽度对齐。 |
| 0Ch | BUFFER\_ADDRESS\_ MSB | 提供缓冲区地址的高 32 位。仅当为大于 32 的地址空间配置 XI DMA 时，才使用此方法。 |
| 10h | CACHE\_USER\_CTL | 位 23：0 – 保留 |
| 位 27：24 – AWCACHE – 缓存类型。此信号提供有关传输的可缓存特征的其他信息。有关不同的解码机制，请参阅 *AMBA AXI 和 ACE 协议规范**[参考文献 3]。*  来自 RX 描述符的 AWCACHE 值在地址周期期间显示在 AWCACHE [3：0] 总线上。此字段的默认值应为 0011。 |
| 位 31：28 – AWUSER – 用于用户定义信息的边带信号。来自 RX 描述符的 AWUSER 值显示在 AWUSER [3：0] 上。AWUSER 值及其解释是用户定义的。您可以通过在链中的所有描述符中编程相同的值来使整个数据包的 AWUSER 保持静态。 |
| 14h | STRIDE\_VSIZE | 位 15：0 – 步幅控制。它是连续“水平”写入的第一个地址之间的地址距离。  写入从缓冲区地址开始并写入 HSIZE 字节，然后跳过 STRIDE-HSIZE 地址并写入 HSIZE 字节，依此类推。这一直持续到写入 VSIZE 为止。上  AXI4-Stream 这在s\_axis\_s2mm\_接口上作为一个连续的包t 接收，并在传输的最后一个数据节拍时以 TLAST 的单个断言终止。 |
| 位 18：16 – 保留 |
| 位 31：19 – 步幅访问的“水平线”数量。可以表示二维视频数据或二维矩阵的大小。每个数据包应接收每个 HSIZE 字节长的 VSIZE 传输数。 |

**RX 描述符字段 *（续）***

|  |  |  |
| --- | --- | --- |
| **地址空间偏移量** | **名字** | **描述** |
| 18h | HSIZE | 位 15：0 – 从连续连续字节地址在每条“水平线”中传输的字节数。当矩阵按行主顺序存储时，可以表示视频行的一部分或矩阵行的一部分。  位 31：16 – 保留 |
| 1Ch | MC\_STS | 多通道状态位。  位 4：0 – TDEST 为数据流提供路由信息。TDEST 值对于整个数据包是静态的。  TDEST 值从传入流捕获并在此字段中更新。 |
| 位 7：5 – 保留 |
| 位 12：8 – TID 提供流标识符。TID 值对于整个数据包是静态的。TID 值从传入流中捕获并在此字段中更新。 |
| 位 15：13 – 保留 |
| 位 19：16 – TUSER – 用于用户定义信息的边带信号。TUSER 值对于整个数据包是静态的。TUSER 值从传入流中捕获并在此字段中更新。 |
| 位 25：20 – 保留 |
| 位 25：24 – 保留 |
| 位 26 – RXEOP – 数据包结束标志。它指示与此描述符关联的缓冲区包含数据包的最后一部分。此标志由 AXI DMA 设置。   * 0 – 不是数据包结束 * 1 – 数据包结束 |
| 位 27 – RXSOP – 数据包标志的开始。它指示与此描述符关联的缓冲区包含数据包的开头。此标志由 AXI DMA 设置。   * 0 – 不是数据包的开始 * 1 – 数据包开始 |
| 位 28 – IE – DMA 由于运行不足或超限而导致的内部错误。   * 0 – 无 DMA 内部错误 * 1 – 检测到 DMA 内部错误。DMA 引擎停止。 |
| 位 29 – SE – DMA 从站错误。如果从属从存储器映射接口读取发出从属错误，则会发生此错误。   * 0 – 无 DMA 从站错误 * 1 – 检测到 DMA 从站错误。DMA 引擎停止。 |

**RX 描述符字段 *（续）***

|  |  |  |
| --- | --- | --- |
| **地址空间偏移量** | **名字** | **描述** |
| 1Ch（续） | MC\_STS | 位 30 – DE – DMA 解码错误。如果地址请求是指向无效地址，则会发生此错误。   * 0 – 无 DMA 解码错误 * 1 – 检测到 DMA 解码错误。DMA 引擎停止。 |
| 位 31 – CMP – 已完成。这向软件指示 DMA 引擎已完成传输。   * 0 – 描述符未完成 * 1 – 描述符完成 |

**笔记：**

1. AWCACHE，AWUSER值对于AXI写入预期很重要。应根据需要在描述符中指定这些值。对于正常操作，AWCACHE 应设置为 0011，而 AWUSER 可以设置为 0000。
2. VSIZE 上的值“0”是非法的，d 会导致多通道 DMA 无法按预期运行。

### AXI DMA 多通道操作

本节介绍 MM2S 端和 S2MM 端的描述符和相关数据的端到端控制和数据流。

#### MM2S

MM2S类似于正常的AXI DMA操作。当MM2S\_CURDESC和

MM2S\_TAILDESC由软件编程时，AXI DMA 会获取一系列描述符并进行处理，直到到达尾部描述符。在 AXI DMA 中，假设 TDEST、TID 和 TUSER 字段在描述符中定义的整个 packet 中保持不变。也就是说，通过 （TDEST、TID、TUSER） 定义的逻辑通道的每个数据包传输在 DMA 传输另一个数据包之前运行完成。尽管多个通道的数据包传输可以交错进行，但在启动后，每个通道都必须运行到完成，然后才能进行另一个传输。在此假设下，您有责任避免死锁情况。如果描述符中的 （TDEST， TID， TUSER） 字段不符合这些假设，则 AXI DMA 不会指示错误条件。由软件来保持一致性。

TX 描述符包含多通道模式下的控制和状态字段。此描述符上没有关于在 tra nsaction 末尾传输的字节数的状态更新。错误信息与失败描述符的当前描述符指针一起在寄存器中捕获。链的传输完成可以通过轮询MM2S\_DMASR的 IDLE 位或通过中断（通过MM2S\_DMACR启用它来了解 ）。

X-参考目标 - 图 2-42

X12595

分散/聚集

DMA Ctrlr

数据移动器

TDEST，TUSER，TID

先进先出

m\_axis\_mm2s

tuser，tdest，tid，tda

ta，tkeep，tvalid，

最后，准备好

m\_axis\_mm2s（读）

m\_axi\_sg

Wr Cntrl

RD Cntrl

*图 2-42* **：MM2S 控制和数据路径流**

#### S2毫米

tdest 信号从传入流中采样。DMA 控制器使用此值读取相应的S2MM\_CURDESC和S2MM\_TAILDESC并将其呈现给分散聚集模块。反过来，这个 SG 模块获取该 tdest 的描述符链。这意味着缓冲区描述符链设置应在数据包到达 S2MM 通道之前完成（针对所有通道）。电流描述符和尾描述符寄存器应在数据包/数据到达之前进行编程。AXI DMA 通过取消断言 tready 来保存流数据， 直到获取相应的描述符。然后，AXI DMA 写入从缓冲区地址开始，一直持续到从流端接收到 tlast。对于特定通道，当数据传输完成时，当前描述符寄存器将加载该链中下一个描述符的地址。这是为了确保正确处理下一个数据包。您必须确保正确设置描述符链，并且正确处理尾部 Descriptor，以避免跳转当前描述符。

TDEST、TID、TUSER 值从传入流中捕获并存储在内部。这些值预计不会在数据包中间更改。在完成该描述符的数据写入到内存映射端后，这些值将与其他状态位一起在链的每个 descr iptor 中更新。

X-参考目标 - 图 2-43

分散/聚集

注册地图

(

新加坡

=1）

TDEST，TUSER，TID

先进先出

m\_axi\_sg

Rd Data

泰斯特

诅咒/

尾部[0：N]

数据移动器

m\_axis\_s2mm

tuser，tdest，tid，tda

ta，tkeep，tvalid，

最后，准备好

m\_axis\_s2mm（写入）

x14589

*图 2-43：* **S2MM 控制和数据路径流**

### 二维传输

在多通道模式下，AXI DMA 支持通过 AXI4 流通道高效传输的二维内存访问模式。

访问模式由描述符字段 HSIZE、VSIZE 和 STRIDE 控制，这些字段允许在（隐式）二维数组内传输子块。HSIZE 在连续“行”子块的起始地址之间指定。对于 2-D 传输，HSIZE、VISZE 和 STRIDE 应按字节对齐。未对齐的 HSIZE、VSIZE 或 STRIDE 值会导致意外行为。

每个读取 （MM2S） 或写入 （S2MM） 传输都由 VSIZE 传输组成，每个传输的大小为 HSIZE。每个连续传输的起始地址是 来自上一个传输的起始地址（最初是数据包传输的 BaseAddr）的 STRIDE 地址。

图 2-44 显示了二维数据格式的示例。

X-参考目标 - 图 2-44

X12594

海泽

VSIZE

跨

跨

缓冲区地址

*图 2-44：* **2-D 数据格式**

#### MM2S

读取从缓冲区地址开始，由 VSIZE 读取突发组成，每个都有 HSIZE 字节。每个读取突发的起始地址是大于上一次突发读取的起始地址的 STRIDE 地址。

示例：缓冲区地址 = 08、VSIZE = 06、HSIZE = 256 字节和步幅 = 512 字节

在这种情况下，读取从缓冲区地址位置 08 开始，并继续读取 HSIZE （256） 字节。 第二行从缓冲区地址+步幅= 512+8 = 520开始。它继续读取 HSIZE （256） 字节。第三条线从 520+512 = 1，032 开始，第四条线从 1032+512 = 1，544 开始。对于 VSIZE 行，以这种模式继续读取。

在 AXI4-Stream 上，它作为一个连续数据包在m\_axis\_mm2s接口上传输，并在传输的最后一个数据节拍中断言 t last 时终止。

#### S2毫米

写入从缓冲区地址开始，由 VSIZE 写入突发组成，每个突发都有 HSIZE 字节。每个写入突发的起始地址是一个 STRIDE 地址，大于前一个 b 次写入的起始地址。在 AXI4-Stream 接口上，该数据包在s\_axis\_s2mm\_接口上作为一个连续数据包接收，并在传输的最后一个数据节拍时以单个断言 tlast 终止。到达数据包的大小应与缓冲区描述符中编程的 wh 匹配。

##### *多通道模式的限制*

* 不支持多通道模式的 S2MM 路径中的描述符队列
* 不支持 S2MM 路径的小数据包大小（4 个或更少数据节拍的背靠背数据包）
* 对于 2-D 访问，地址和 Hsize 必须与内存映射数据宽度对齐。

### 多通道寄存器映射 （SG = 1）

修改寄存器映射，以支持 S2MM 路径上最多 16 个 TDEST。

X-参考目标 - 图 2-45

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 00 | MM2S\_DMACR |  | C0 | 保留 |  | 180 | | 保留 |
| 04 | MM2S\_DMASR | C4 | 保留 | 184 | | 保留 |
| 08 | MM2S\_CURDESC0 | C8 | 保留 | 188 | | 保留 |
| 0C | MM2S\_CURDESC0\_MSB | cc | 保留 | 18C | | 保留 |
| 10 | MM2S\_TAILDESC0 | D0 | S2MM\_CURDESC4 | 190 | | S2MM\_CURDESC10 |
| 14 | MM2S\_TAILDESC0\_MSB | D4 | S2MM\_CURDESC4\_MSB | 194 | | S2MM\_CURDESC10\_MSB |
| 18d | 保留 | D8 | S2毫米-尾部4 | 198 | | S2毫米尾部10 |
| 1C | 保留 | 直流 | S2MM\_TAILDESC4\_MSB | 19C | | S2MM\_TAILDESC10\_MSB |
| 20 | 保留 | E0 | 保留 | 1A0 | | 保留 |
| 24 | 保留 | E4 | 保留 | 1A4 | | 保留 |
| 28 | 保留 | E8 | 保留 | 1A8 | | 保留 |
| 2C | SG\_CTL | 电子商务 | 保留 | 1交流电 | | 保留 |
| 30 | S2MM\_DMACR | F0 | S2MM\_CURDESC5 | 1B0 | | S2MM\_CURDESC11 |
| 34 | S2MM\_DMASR | F4 | S2MM\_CURDESC5\_MSB | 1B4 | | S2MM\_CURDESC11\_MSB |
| 38 | S2MM\_CURDESC0 | F8 | S2MM\_TAILDESC5 | 1B8 | | S2MM\_TAILDESC11 |
| 3C | S2MM\_CURDESC0\_MSB | FC | S2MM\_TAILDESC5\_MSB | 1BC | | S2MM\_TAILDESC11\_MSB |
| 40 | S2MM\_TAILDESC0 | 100 | 保留 | 1C0 | | 保留 |
| 44 | S2MM\_TAILDESC0\_MSB | 104 | 保留 | 1C4 | | 保留 |
| 48 | 保留 | 108 | 保留 | 1C8 | | 保留 |
| 4C | 保留 | 10C | 保留 | 1CC | | 保留 |
| 50 | 保留 | 110 | S2MM\_CURDESC6 | 1天0 | | S2MM\_CURDESC12 |
| 54 | 保留 | 114 | S2MM\_CURDESC6\_MSB | 1天4 | | S2MM\_CURDESC12\_MSB |
| 58 | 保留 | 118 | S2MM\_TAILDESC6 | 1天8 | | S2MM\_TAILDESC12 |
| 5C | 保留 | 11C | S2MM\_TAILDESC6\_MSB | 1直流 | | S2MM\_TAILDESC12\_MSB |
| 60 | 保留 | 120 | 保留 | 1 | E0 | 保留 |
| 64 | 保留 | 124 | 保留 | 1E4 | | 保留 |
| 68 | 保留 | 128 | 保留 | 1E8 | | 保留 |
| 6C | 保留 | 12C | 保留 | 1EC | | 保留 |
| 70 | S2MM\_CURDESC1 | 130 | S2MM\_CURDESC7 | 1F0 | | S2MM\_CURDESC13 |
| 74 | S2MM\_CURDESC1\_MSB | 134 | S2MM\_CURDESC7\_MSB | 1F4 | | S2MM\_CURDESC13\_MSB |
| 78 | S2MM\_TAILDESC1 | 138  13C  140  144  148  14C  150 | S2MM\_TAILDESC7 | 1F8 | | S2MM\_TAILDESC13 |
| 7C | S2MM\_TAILDESC1\_MSB | S2MM\_TAILDESC7\_MSB | 1FC | | S2MM\_TAILDESC13\_MSB |
| 80 | 保留 | 保留 | 200 | | 保留 |
| 84 | 保留 | 保留 | 204 | | 保留 |
| 88 | 保留 | 保留 | 208 | | 保留 |
| 8C | 保留 | 保留 | 20摄氏度 | | 保留 |
| 90 | S2MM\_CURDESC2 | S2MM\_CURDESC8 | 210 | | S2MM\_CURDESC14 |
| 94 | S2MM\_CURDESC2\_MSB | 154 | S2MM\_CURDESC8\_MSB | 214 | | S2MM\_CURDESC14\_MSB |
| 98 | S2MM\_TAILDESC2 | 158 | S2MM\_TAILDESC8 | 218 | | S2MM\_TAILDESC14 |
| 9C | S2MM\_TAILDESC2\_MSB | 15C  160  164  168 | S2MM\_TAILDESC8\_MSB | 21C | | S2MM\_TAILDESC14\_MSB |
| 答0 | 保留 | 保留 | 220 | | 保留 |
| 答4 | 保留 | 保留 | 224 | | 保留 |
| 血型 | 保留 | 保留 | 228 | | 保留 |
| 交流 | 保留 | 16C | 保留 | 22摄氏度 | | 保留 |
| B0 | S2MM\_CURDESC3 | 170 | S2MM\_CURDESC9 | 230 | | S2MM\_CURDESC15 |
| B4 | S2MM\_CURDESC3\_MSB | 174  178  17C | S2MM\_CURDESC9\_MSB | 234 | | S2MM\_CURDESC15\_MSB |
| B8 | S2MM\_TAILDESC3 | S2MM\_TAILDESC9 | 238 | | S2MM\_TAILDESC15 |
| 公元前 | S2MM\_TAILDESC3\_MSB | S2MM\_TAILDESC9\_MSB | 23C | | S2MM\_TAILDESC15\_MSB |

X12591

*图 2-45：* **SG = 1 中多通道支持的寄存器映射**



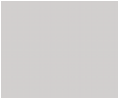
*第 3 章* 核心设计

## 典型系统互连

AXI DMA 内核设计为通过用户系统中的 AXI 互连进行连接。典型的 MicroBlaze™ 处理器配置如图 3-1 所示。系统微处理器可通过 AXI4-Lite 接口访问 AXI DMA。inte磨碎的分散/收集引擎从系统内存中获取缓冲区描述符，然后协调 AXI IP 和 DDRx 之间的主数据传输。 可选的控制和状态流提供与数据包相关的信息，例如校验和卸载控制/status，往返于基于以太网的 IP。AXI DMA 内核的双中断输出被路由到系统中断控制器。

X-参考目标 - 图 3-1

$;，'0$



00

6

5

热镀锌

LFUR%OD]H

0

URFHVVRU

3

''5[

0

HPRU\&RQWUR

呜



3



3

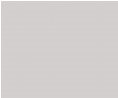


3

6\*

5

:



00

6

:

乌尔瓦

00

6

6

WUP

德维乌

0

6

WUP

$;，（WKHUQHW

WDWXV

6

WUP

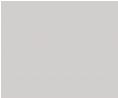
6

德维乌

0

WUP

6



)

WDWXV，

6

6

奥迪

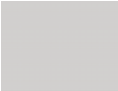
WUP

6

&RQWURO，）

，QWHUUXSW

&RQWUROOHU



$;/LWH

6B，QWU2XW

00

6

00B，QWU2XW

$;，，QWHUFRQQHFW

$;，，QWHUFRQQHFW

$;，，QWHUFRQQHFW

$;，，QWHUFRQQHFW

$;6WUHDP

$;6WUHDP

$;6WUHDP

$;6WUHDP

;

3

吴和

6WUP

*图 3-1：* **典型的 MicroBlaze 处理器系统配置（AXI 以太网）**

AXI DMA 内核还可以连接到用户系统，而不是使用基于以太网的 AXI IP。控制和状态流是可选的，只能与基于以太网的 IP 核一起使用。

***注意：***在没有任何设置的情况下（即，在编程运行之前），AXI DMA 将在 接收四拍流数据后将s\_axis\_s2mm\_tready信号拉低。这将限制输入数据流。要获得最少的限制，请确保将 AXI DMA 设置为在实际 data 到达之前运行。

## 时钟

有四个时钟输入：

* MM2S 接口m\_axi\_mm2s\_aclk
* 用于 S2MM 接口的m\_axi\_s2mm\_aclk
* AXI4-Lite 控制接口的 s\_axi\_lite\_aclk
* 分散聚集接口的m\_axi\_sg\_clk

AXI DMA 提供两种时钟操作模式：异步和同步。设置启用异步时钟可启用异步模式并创建四个时钟域。这允许高性能用户以比 DMA 控件（例如，AXI4-Lite 接口、SG 引擎、DMA 控制器）更高的时钟运行主数据路径，从而帮助 FPGA 放置和时序。

在同步模式下，所有逻辑都在单个时钟域中运行。m\_axi\_sg\_aclk，m\_axi\_mm2s\_aclk和m\_axi\_s2mm\_aclk必须绑定到同一源，s\_axi\_lite\_aclk可以连接到较慢的时钟。在异步模式下，时钟可以异步运行，但s\_axi\_lite\_aclk 必须小于或等于m\_axi\_sg\_aclk，m\_axi\_sg\_aclk必须小于或等于较慢的m\_axi\_mm2s\_aclk或m\_axi\_s2mm\_aclk。

异步模式下信号集与其对应时钟之间的关系如表3-1所示。

*表 3-1：* **异步模式时钟分布**

|  |  |  |
| --- | --- | --- |
| **时钟源** | **I/O 端口（启用分散收集）** | **I/O 端口（禁用分散收集）** |
| s\_axi\_lite\_aclk | 所有s\_axi\_lite\_\*信号mm2s\_introut s2mm\_introut axi\_resetn | 所有s\_axi\_lite\_\*信号mm2s\_introut s2mm\_introut axi\_resetn |
| m\_axi\_sg\_aclk | 所有m\_axi\_sg\_\*信号 | 不适用 |
| m\_axi\_mm2s\_aclk | 所有 m\_axi\_mm2s\_\* 信号 所有 m\_axis\_mm2s\_\* 信号 mm2s\_prmry\_reset\_out\_n mm2s\_cntrl\_reset\_out\_n | 所有m\_axi\_mm2s\_\*信号 所有m\_axis\_mm2s\_\*信号mm2s\_prmry\_reset\_out\_n |
| m\_axi\_s2mm\_aclk | 所有m\_axi\_s2mm\_\*信号 所有s\_axis\_s2mm\_\*信号s2mm\_prmry\_reset\_out\_n s2mm\_sts\_reset\_out\_n | 所有m\_axi\_s2mm\_\*信号 所有s\_axis\_s2mm\_\*信号s2mm\_prmry\_reset\_out\_n |

## 重 置

axi\_resetn信号需要至少置位16个最慢的时钟周期，并且需要同步到s\_axi\_lite\_aclk。

## 编程顺序

### 直接寄存器模式（简单 DMA）

直接寄存器模式（分散收集引擎已禁用）提供了一种配置，用于在 MM2S 和 S2MM 通道上执行简单的 DMA 传输，这需要较少的 FPGA 资源利用率。传输是通过访问DMACR，地址上的源或目的地以及长度寄存器来启动的。传输完成后，将进行 DMASR。IOC\_Irq断言关联的通道，如果启用，则会生成中断输出。

MM2S 通道的 DMA 操作按以下顺序设置和启动：

1. 通过将运行/停止位设置为 1 （MM2S\_DMACR 来启动 MM2S 通道运行。RS = 1）。停止的位（DMASR.已停止）应取消断言，指示 MM2S 通道正在运行。
2. 如果需要，通过写入 1 到 MM2S\_DMACR 来启用中断。IOC\_IrqEn和

MM2S\_DMACR。Err\_Irq恩。将 AXI DMA 配置为直接寄存器模式时，不使用延迟中断、延迟计数和阈值计数。

1. 将有效的源地址写入MM2S\_SA寄存器。如果为大于 32 的地址空间配置了 AXI DMA，则对MSB 寄存器MM2S\_SA进行编程。如果未为数据重新对齐配置 AXI DMA，则必须对齐有效的地址，否则会出现未定义的结果。被视为对齐或不对齐的内容基于流数据宽度。在Mi cro模式下配置AXI\_DMA时，您有责任指定正确的地址。Micro DMA不负责4K边界。

例如，如果内存映射数据宽度 = 32，则数据位于字偏移量（32 位偏移量）处（即0x0、0x4、0x8、0xC等）时，数据将对齐。如果启用了 DRE 并且流数据宽度< 128，则源地址可以是任意字节偏移量。

1. 在MM2S\_LENGTH寄存器中写入要传输的字节数。写入的值为零不起作用。非零值会导致在 MM2S AXI4 接口上读取MM2S\_LENGTH字节数，并从 MM2S AXI4 流接口传输出去。MM2S\_LENGTH登记册必须写在最后。所有其他MM2S寄存器可以按任意顺序写入。对于微型 DMA，此值不能超过 [Burst\_length \*（内存映射数据宽度）/8]。

S2MM 通道的 DMA 操作按以下顺序设置和启动：

1. 通过将运行/停止位设置为 1 （S2MM\_DMACR 来启动运行的 S2MM 通道。RS = 1）。停止的位（DMASR.已停止）应取消断言，指示 S2MM 通道正在运行。
2. 如果需要，通过写入 1 到 S2MM\_DMACR.IOC\_IrqEn 和

S2MM\_DMACR。Err\_IrqEn。将 AXI DMA 配置为直接寄存器模式时，不使用延迟中断、延迟计数和阈值计数。

1. 将有效的目标地址写入S2MM\_DA寄存器。如果 AXI DMA 配置为大于 32 的广告着装空间，请对 S2MM\_DA MSB 寄存器进行编程。
2. 如果未为 AXI DMA 配置数据重新对齐，则必须对齐有效的地址，否则会出现未定义的结果。什么被认为是对齐或未对齐的是基于流数据widt h。

例如，如果内存映射数据宽度 = 32，则如果数据位于字偏移量（32 位偏移量），即0x0、0x4、0x8、0xC等，则数据将对齐。如果启用了 DRE，并且

流数据宽度< 128，则目标地址可以是任何字节 偏移量。

1. 将接收缓冲区的长度（以字节为单位）写入S2MM\_LENGTH寄存器中。值为零不起作用。非零值会导致在 S2MM AXI4 接口上写入 S2MM AXI4 流接口上接收的字节数。大于或等于最大接收数据包的值必须写入S2MM\_LENGTH。小于接收字节数的接收缓冲区长度值会产生未定义的结果。在微模式下配置 AXI DMA 时，此值应与S2MM AXI4 流接口上接收的字节数完全相同。S2MM\_LENGTH寄存器必须写在最后。所有其他S2MM寄存器可以按任意顺序写入。

### 分散/聚集模式

AXI DMA 操作需要一个内存驻留数据结构，用于保存要执行的 DMA 操作的列表。此指令列表被组织到所谓的描述符链中。每个描述符都有一个指向要处理的下一个描述符的指针。然后，链中的最后一个描述符指向链中的第一个描述符。

分散收集操作允许由多个描述符描述数据包。此功能的典型用途是允许从内存中的某个位置存储或提取标头，并从另一个位置存储或提取有效负载数据。Softw是利用这一点可以提高吞吐量的应用程序。要描述缓冲区描述符链中的数据包，请使用帧起始位 （TXSOF） 和帧结束位 （TXEOF）。当 DMA 获取设置了 TXSOF 位的描述符时， 将触发数据包的开始。数据包继续获取后续描述符，直到获取设置了 TXEOF 位的描述符。

在接收 （S2MM） 通道上，当开始接收数据包时，AXI DMA 使用 RXSOF 标记描述符， 向软件指示与此描述符关联的数据缓冲区包含数据包的开头。如果接收的数据包的字节计数长于描述符中指定的字节计数，则下一个描述符缓冲区用于 存储接收数据包的其余部分。此获取和存储过程将继续，直到传输整个接收数据包。接收数据包末尾时正在处理的描述符由 AXI DMA 标记，RXEOF=1。这向软件指示与此描述符关联的缓冲区包含数据包的末尾。

每个描述符的状态字段包含该特定描述符实际传输的字节数。该软件可以通过从 RXSOF 描述符通过 d描述符链到 RXEOF 描述符来确定接收数据包传输的总字节数。分散聚集继续获取一个额外的描述符和存储。此过程在很大程度上提高了 DMA 性能。

分散收集操作从设置控制寄存器和描述器指针开始。

使用以下顺序设置和启动 MM2S 通道的 DMA 操作：

1. 将起始描述符的地址写入当前描述符寄存器。如果 AXI DMA 配置为地址空间大于 32，则 还要对当前描述符的 MSB 32 位进行编程。
2. 通过将运行/停止位设置为 1 （MM2S\_DMACR 来启动 MM2S 通道运行。RS =1）。已停止位 （DMASR.已停止）应取消断言，指示 MM2S 通道正在运行。
3. 如果需要，通过 写入 1 到 MM2S\_DMACR 来启用中断。IOC\_IrqEn和MM2S\_DMACR。Err\_IrqEn。
4. 将有效地址写入尾部描述符寄存器。如果 AXI DMA 配置为地址空间大于 32，则还要对尾部描述符的 MSB 32 位进行编程。
5. 写入 尾部描述符寄存器会触发 DMA 开始从内存中获取描述符。在多通道配置的情况下，当数据包到达 S2MM 通道时，描述符的获取开始。
6. 处理获取的描述符， 从内存中读取数据，然后输出到 MM2S 流通道。

使用以下顺序设置和启动 S2MM 通道的 DMA 操作：

1. 将起始描述符的地址写入当前描述符寄存器。如果 AXI DMA 配置为地址空间大于 32 ，则还要对当前描述符的 MSB 32 位进行编程。
2. 通过将运行/停止位设置为 1 （S2MM\_DMACR 来启动运行的 S2MM 通道。RS =1）。停止的位（DMASR.停止）应取消断言，指示S2MM 通道正在运行。
3. 如果需要，通过写入 1 到 S2MM\_DMACR 来启用中断。IOC\_IrqEn和S2MM\_DMACR。Err\_IrqEn。
4. 将有效地址写入尾部描述符寄存器。如果 AXI DMA 配置为地址空间大于 32，则当前描述符的 MSB 32 位也进行编程。
5. 写入尾部描述符寄存器会触发 DMA 开始从内存中获取描述符。
6. 处理提取的描述符，并从 S2MM 流通道接收的任何数据写入 内存。

### 循环 DMA 模式

AXI DMA 可以通过对缓冲区描述符 （BD） 链设置进行某些更改，在循环模式下运行。在循环模式下，DMA 会不间断地获取和处理相同的 BD。DMA 将继续提取和处理，直到停止或重置。要启用循环操作，应如图3-2所示设置BD链。

X-Ref Target - 图 3-2

第一BD （0x40）

BD1

BD2

BDn

X14564

尾部BD （0x1C0）

*图 3-2：* **BD 链**

在此设置中，尾部BD指向第一个BD。尾部描述符寄存器没有任何用途，仅用于触发 DMA 引擎。遵循分散/聚集模式中提到的相同编程顺序。确保已设置控制寄存器中的循环位。

使用一些不属于BD链的值对尾部描述符寄存器进行编程。比如 说0x50。

对尾部描述符寄存器进行编程后，DMA 开始获取和处理 BD（ 以环形方式设置），直到 DMA 停止或重置。



# *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、综合和实现步骤。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： 利用 IP 进行设计* （UG896） [参考文献 1]
* *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： 使用 IP 集成器设计 IP* 子系统 （UG994） [参考文献 5]
* *Vivado 设计套件用户指南： 逻辑仿真* （UG900） [参考文献 6]

## 自定义和生成核心

AXI DMA 可在 Vivado IP 目录中的以下位置找到。

* **AXI\_Infrastructure、Communication\_&\_Networking\以太网**
* **Embedded\_Processing\AXI\_Infrastructure\DMA**

要访问 AXI DMA，请执行以下操作：

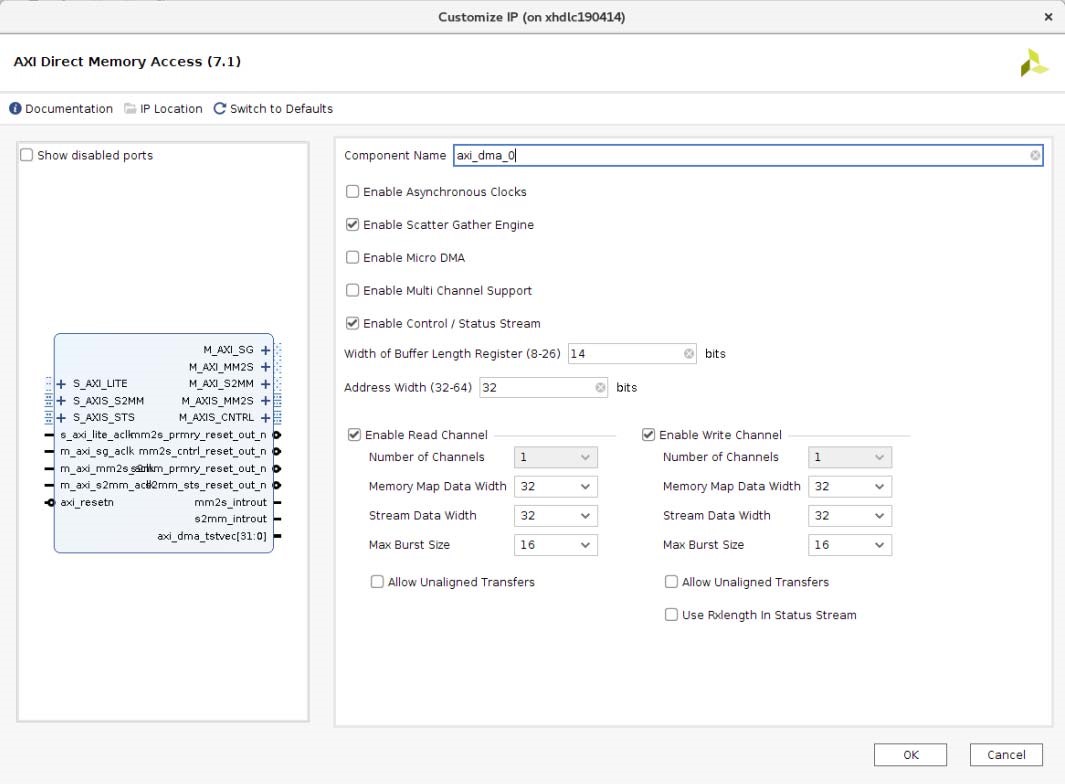
1. 使用 Vivado 设计工具打开现有项目或创建新项目。
2. 打开 IP 目录并导航到任何分类。
3. 双击“**AXI 直接内存访问**”以打开“AXI DMA 自定义 IP”窗口。

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 1] 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集成设计环境（IDE）的图示。此布局可能与当前版本不同。

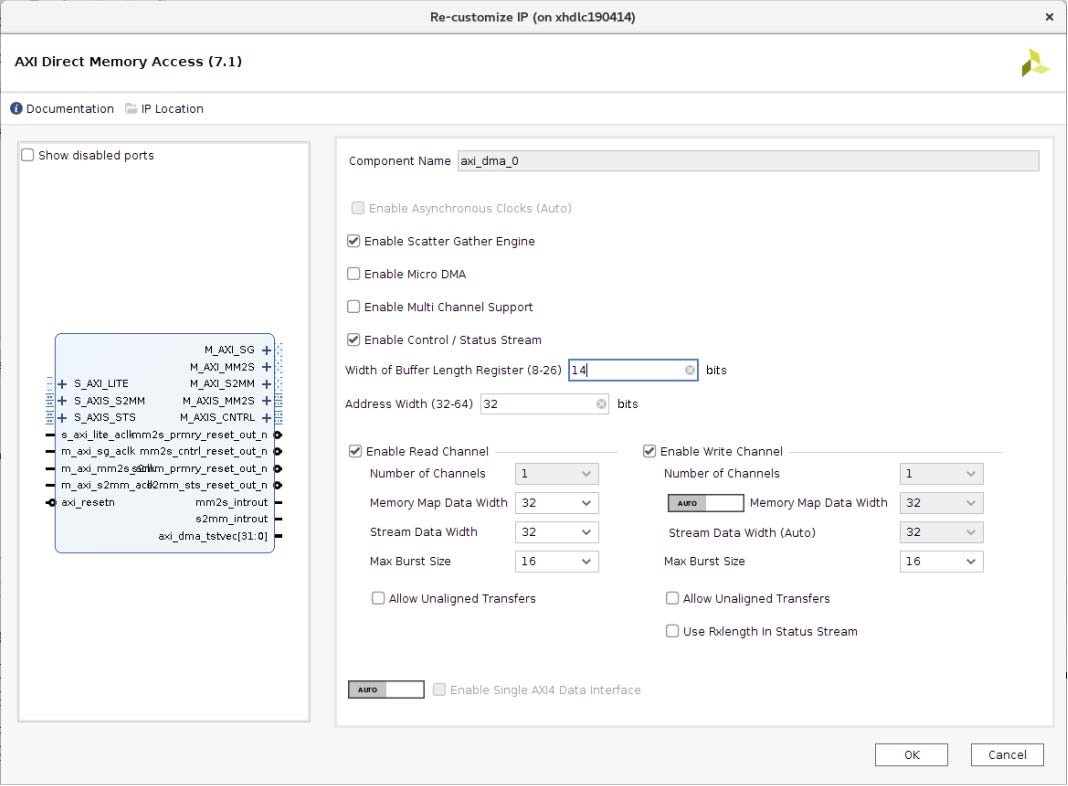
如果要在 Vivado IP 集成器中定制和生成内核，请参阅*《Vivado 设计套件用户指南：使用 IP 集成器设计 IP 子系统* （UG994） [参考文献 5]》，了解详细信息。如本节所述，Vivado IDE 可能会在验证或生成设计时自动计算某些配置值。您可以在 su 完成 validate\_bd\_design 命令后查看参数值 。

X-参考目标 - 图 4-1



*图 4-1* **：自定义 IP 选项**

X-参考目标 - 图 4-2



*图 4-2：* **IP 集成商**

### 字段说明

#### 组件名称

为核心生成的输出文件的基本名称。名称必须以字母开头，并且可以由以下任何字符组成：a 到 z、0 到 9 和“\_”。

#### 启用异步时钟

此设置提供了相互异步操作 MM2S接口间m\_axi\_mm2s\_aclk、S2MM 接口m\_axi\_s2mm\_aclk、AXI4-Lite 控制接口s\_axi\_lite\_aclk和分散收集接口m\_axi\_sg\_aclk的能力。启用异步时钟时，s\_axi\_lite\_aclk的频率必须小于或等于 m\_axi\_sg\_aclk。同样，m\_axi\_sg\_aclk必须小于或等于m\_axi\_mm2s\_aclk和m\_axi\_s2mm\_aclk的较慢值。在同步模式下，所有时钟输入应连接到相同的时钟信号。此参数在Vivado IP 积分器中根据连接到axi\_dma的时钟自动设置。

**提示：**

*当 IP 在 Vivado IP 积分器中使用时，会自动设置此参数。*

#### 启用散点收集引擎

选中此选项将启用分散收集模式操作，并在 AXI DMA 中包含分散收集引擎。取消选中此选项将启用直接寄存器模式操作，但不包括 AXI DMA 中的分散收集引擎。禁用分散收集引擎会导致分散/聚集引擎的所有输出端口绑定为零，并且所有输入端口保持打开状态。

#### 启用微型 DMA

选中此选项会生成资源计数较少的高度优化的 DMA。此设置可用于传输非常少量 dat a 的应用程序。 根据所选配置对 DMA 进行编程。例如，每个事务或每个BD可以传输的最大字节数不能超过以下内容：

MMap Data\_width \* Burst\_length/8。

同样，在此模式下也不会实现 4K 边界检查，将 寻址限制为突发边界。

#### 缓冲区长度寄存器的宽度

此整数值指定用于分散/收集描述符中传输的控制字段缓冲区长度和状态字段字节的有效位数。 它还指定启用“使用 Rxlength”时状态流 App4 字段的 RX 长度中的有效位数。对于直接寄存器模式，它指定有效位数在

MM2S\_LENGTH和S2MM\_LENGTH寄存器。长度宽度与分散/收集描述符中指定的字节数或在 App4.RxLength、MM2S\_LENGTH 或 S2MM\_LENGTH 中指定的字节数直接相关。字节数等于 2长度宽度。因此，长度宽度为 26 的字节数为 67，108，863 字节。对于多通道模式，此值应设置为 23。

***地址宽度 （32 - 64）***

指定地址空间的宽度。它可以是 32 到 64 之间的任何值。

#### 启用多通道 DMA

***注意：***多通道功能将很快停止使用。有关多通道的信息，请参阅 AXI 多通道直接内存访问 （PG288） *[参考文献 12]。*

选中此选项可启用 DMA 的多通道功能，并允许您为 MM2S 和 S2MM 通道选择通道数。 有关详细信息，请参阅第 2 章中的多通道 DMA 支持。

#### 启用控制/状态流

选中此选项将启用 AXI4 控制和状态流。AXI4 控制流允许将与 MM2S 通道关联的用户应用程序元数据传输到目标 IP。MM2S 的用户应用程序字段 0 到 4 分散/收集帧的起点

（SOF） 描述符 Transmit 帧起始 （TXSOF =1） 在m\_axis\_mm2s\_cntrl流接口上传输，同时在m\_axis\_mm2s流接口上传输关联的数据包 。AXI4 状态流允许从目标 IP 接收与 S 2MM 通道关联的用户应用程序元数据。接收的状态数据包填充 S2MM 散点/收集帧结束 （EOF） 描述符的用户应用程序字段 0 到 4。这是与数据包末尾关联的描述符。此条件由更新描述符的状态字中的帧接收结束 （RXEOF = 1） 指示。

#### 启用读取通道选项

以下小节介绍仅影响 AXI 直接内存访问 （DMA） 内核的 MM2S 通道的选项。

##### 启用通道

此选项启用或禁用 MM2S 通道。启用 MM2S 通道允许从内存到 AXI4 流进行读取传输。禁用 MM2S 通道会从 AXI DMA 内核中排除逻辑。MM2S 通道的输出为零， AXI DMA 忽略输入。

#### 通道数

此选项指定从 1 到 16 的通道数。

##### 内存映射数据宽度

AXI MM2S 内存映射读取数据总线的数据宽度（以位为单位）。有效值为 32、64、128、256、512 和 1，024。

##### 流数据宽度

D ata 宽度（以位为单位）的 AXI MM2S AXI4 流数据总线。此值必须等于或小于内存映射数据宽度。有效值为 8、16、32、64、128、512 和 1，024。

##### 最大突发大小

突发分区粒度设置。此设置指定MM2S 的 AXI4 内存映射端突发周期的最小大小。有效值为 2、4、8、16、32、64、128 和 256。

##### 允许未对齐的传输

启用或禁用 MM2S 数据重新对齐引擎 （DRE）。选中后，DRE 将启用，并允许数据重新对齐到 MM2S 内存映射数据路径上的字节（8 位）级别。对于 MM2S 通道，从内存中读取数据。如果启用了 DRE，则数据读取可以从任何缓冲区地址字节偏移量开始，并且读取的数据将对齐，以便第一个字节 read 是 AXI4 流上的第一个有效字节。

***注意：*** 如果对相应通道禁用了 DRE，则不支持未对齐的缓冲区、源或目标地址。禁用 DRE 的未对齐地址会产生未定义的结果。DRE Support 仅适用于 512 位及以下的 AXI4 流数据宽度设置。

#### 启用写入通道选项

这些选项仅影响 AXI DMA 内核的 S2MM 通道。

##### 启用通道

此设置启用或禁用 S2MM 通道。启用 S2MM 通道允许从 AXI4 流到内存进行写传输。禁用 S2MM 通道会从 AXI DMA 内核中排除逻辑。S2MM 通道的输出与零绑定，AXI DMA 忽略 inpu ts。

**通道数**

此选项允许您从 1 到 16 选择多个通道。

##### 内存映射数据宽度

AXI S2MM 内存映射写入数据总线的数据宽度（以位为单位）。有效值为 32、64、128、256、512 和 1，024。

**提示：***在 Vivado IP 集成器中，此参数会根据流接口的数据宽度自动设置。通过将开关更改为“手动”来*更新此参数 r。

##### 流数据宽度

AXI S2MM AXI4 流数据总线的数据宽度（以位为单位）。此值必须等于或小于内存映射数据宽度。有效值为 8、16、32、64、128、512 和 1，024。

**提示：** *在 Vivado IP 积分器中使用 IP 时，此参数将根据与s\_axis\_s2mm接口的连接自动设置。*

#### 最大突发大小

此设置指定 S2MM 通道的 AXI4 内存映射端突发周期的最大大小。在其他wo rds 中，此设置指定突发映射的粒度。有效值为 2、4、8、16、32、64、128 和 256。

##### 允许未对齐的传输

启用或禁用 S2MM 数据重新对齐引擎 （DRE）。选中后，DRE 将启用，并允许将数据重新对齐到 S2MM 内存映射数据路径上的字节（8 位）级别。对于 S2MM 通道，数据被写入内存。如果启用了 DRE，则写入可以从任何缓冲区地址字节偏移量开始，并且写入数据将对齐，以便将 S2MM AXI4-Stream 上接收的第一个有效字节写入指定的未对齐地址偏移量。

***注意：*** 如果为相应通道禁用了 DRE，则不支持未对齐的缓冲区、源或目标地址。禁用 DRE 的未对齐地址会产生未定义的结果。DRE 支持仅适用于 512 位及以下的 AXI4 流数据宽度设置。

##### 在状态 Str 中使用 RxLength

如果启用了控制/状态流，则选中此选项将允许 AXI DMA 使用状态数据包的 App4 字段中的 S2MM 目标 IP 提供的接收长度字段。这为 AXI DMA 提供了预先确定的接收字节计数，从而允许 AXI DMA 命令要传输的确切字节数。

此选项为需要更大吞吐量的系统提供了更高的带宽解决方案。在此配置中，S2MM 目标 IP 可以提供状态数据包 APP4 的接收长度字段中指定的所有数据字节。

##### 启用单个 AXI4 数据接口

此选项仅在 Vivado IP 集成器中使用时才适用。可以使用此选项将两个 AXI4 接口（MM2S 和 S2MM）合并为一个接口。此选项不会影响源或性能。

### 用户参数

表 4-1 显示了 Vivado IDE 中的字段与用户参数之间的关系（可在工具命令语言 （Tcl） 控制台中查看）。

*表 4-1：* **Vivado IDE 参数与用户参数的关系**

|  |  |  |
| --- | --- | --- |
| **Vivado IDE 参数** | **用户参数** | **默认值** |
| 启用散点收集引擎 | c\_include\_sg | 1 |
| 启用多渠道支持 | c\_enable\_multi\_channel | 0 |
| 通道数 （MM2S） | c\_num\_mm2s\_channels | 1 |
| 通道数 （S2MM） | c\_num\_s2mm\_channels | 1 |
| 缓冲区长度寄存器的宽度 | c\_sg\_length\_width | 14 |
| 启用异步时钟 | c\_prmry\_is\_aclk\_async | 0 |
| 启用控制/状态流 | c\_sg\_include\_stscntrl\_strm | 1 |
| 启用微型 DMA | c\_micro\_dma | 0 |
| 启用读取通道 | c\_include\_mm2s | 1 |
| 内存映射数据宽度 （MM2S） | c\_m\_axi\_mm2s\_data\_width | 32 |
| 流数据宽度 （MM2S） | c\_m\_axis\_mm2s\_tdata\_width | 32 |
| 允许未对齐的传输 （MM2S） | c\_include\_mm2s\_dre | 0 |
| 最大突发尺寸 （MM2S） | c\_mm2s\_burst\_size | 16 |
| 启用写入通道 | c\_include\_s2mm | 1 |
| 在状态流中使用 Rxlength | c\_sg\_use\_stsapp\_length | 0 |
| 内存映射数据宽度 （S2MM） | c\_m\_axi\_s2mm\_data\_width | 32 |
| 流数据宽度 （S2MM） | c\_s\_axis\_s2mm\_tdata\_width | 32 |
| 允许未对齐的传输 （S2MM） | c\_include\_s2mm\_dre | 0 |
| 最大突发尺寸 （S2MM） | c\_s2mm\_burst\_size | 16 |
| 地址宽度 （32-64） | c\_addr\_width | 32 |

### 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 1]。

## 约束核心

必要的 XDC 约束随 Vivado 设计套件中的内核生成一起提供。

**必需的约束**

本节不适用于此 IP公司。

**器件、封装和速度等级选择**

本节不适用于此 IP 核。

**时钟频率**

本节不适用于此 IP 核。

**时钟管理**

本节不适用于此 IP 核。

**时钟放置**

本节不适用于此 IP 核。

**银行业**

本节不适用于此 IP 核。

**收发器放置**

本节不适用于此 IP 核。

### I/O 标准 and 放置

本节不适用于此 IP 核。

## 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 6]。

**重要说明：** *对于面向 7 系列或 Zynq-7000 设备的内核，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，请参见 *Vivado 设计套件用户指南：*

*使用 IP 进行设计* （UG896） [参考文献 1]。



*第5章* 示例设计

本章包含有关 Vivado® 设计套件中提供的示例设计的信息。

顶部模块实例化内核和示例设计的所有组件，这些组件需要在硬件中实现设计，如图 5-1 所示。这包括混合模式时钟管理器 （MMCME2）、寄存器配置、数据生成器和数据检查器模块。

X-参考目标 - 图 5-1

读取数据生成器

被测器

时钟发生器

寄存器配置

Componetname>\_exdes.vhd

<

)

返回页首

(

AXI4 RD

clock\_in

重置

写入数据检查器

AXI4 WR

腋石

读取数据检查器

写入数据生成器

安讯士研发

安讯士 WR

到示例

设计

模块

axi4\_write\_master.vhd

clock\_gen.vhd

axis\_data\_read.vhd

axis\_write\_master.vhd

axi\_s2mm\_read.vhd

SG BRAM

阅读 CTRL 检查器

axis\_ctrl\_read.vhd

写入数据生成器

axis\_sts\_master.vhd

腋石

AXI SG

做

地位

X13594

*图 5-1：* **示例设计框图**

此示例设计演示了 DUT 的 AXI4-Lite、AXI4 和 AXI4-Stream 接口上的事务。

**时钟发生器**：MMCME2 用于生成示例设计的时钟。当 DUT 处于同步模式时，MMCME2为示例设计中的所有 AXI 接口生成 100 MHz clo ck。在异步模式下，MMCME2 为 AXI4-Lite 接口生成一个 50 MHz 时钟，为 AXI4 和 AXI4 流接口生成一个 100 MHz 时钟 。

示例设计的 DUT 和其他模块在复位状态下保持 pt，直到 MMCME2 锁定。

**寄存器配置模块：该模块按照**第 3 章编程序列中的编程顺序中所述配置 DUT 寄存器。此模块是一个 AXI 流量生成器模块，用于对寄存器进行编程。有关详细信息，请参阅 *AXI 流量生成器 LogiCORE IP* （PG125） [参考文献 7]。

**读取路径生成器：** 这使用 AXI 块 RAM，该 RAM 在 MMCME2 锁定后填充（具有固定数量的传输）。MM2S 通道读取AXI 块 RAM 并将数据传输到 AXI4 流接口。

**读取路径检查器：** 此模块检查在 MM2S AXI4 流接口上传输的数据。

**读取路径 CTRL 检查器**：此模块检查在 MM2S AXI4 流控制接口上传输的数据 。

**写入路径生成器**：配置写入 （S2MM） 通道后，此模块驱动 S2MM AXI4-Stream 接口上的事务（具有固定数量的传输）。 **写入路径 STS 生成器**：此模块生成 S2MM STS 数据流。

**写入路径检查器**：此模式检查在 AXI4 接口上接收的数据。在 AXI4 接口上接收的数据也会写入另一个 AXI 块 RAM。

MMCME2 锁定后不久，测试将开始。完成所有事务后，“完成”引脚断言为“高”。Simi larly 状态引脚在数据完整性检查成功时被断言为高。这两个引脚可以连接到LED以了解测试状态。

## 实现示例设计

按照第 4 章 “设计流程步骤 s ”中描述的步骤操作后，按如下方式实现示例设计：

1. 右键单击“层次结构”窗口中的内核，然后选择“ **打开 IP 示例设计**”。
2. 将弹出一个新窗口，要求您为示例设计指定目录。选择新目录，或保留默认目录。

将在所选目录中自动创建一个新项目，并在新的 Vivado IDE 窗口中打开。

1. 在流导航器（左侧窗格）中，单击 **运行实施** 并按照说明进行操作。

表 5-1、表 5-2 和表 5-3 描述了示例设计、仿真和约束目录中的文件。

在当前项目目录中，名称为

将创建<component\_name>\_example并将文件传递到该目录中。此目录及其子目录包含创建 AXI DMA 控制器示例设计所需的所有源文件。

表 5-1 显示了作为示例设计一部分的文件。

*表 5-1* **：示例设计目录**

|  |  |
| --- | --- |
| **名字** | **描述** |
| <component\_name>\_exdes.vhd | 示例设计的顶级 HDL 文件。 |
| axi\_lite\_sm.vhd | 注册示例设计的配置文件。  （默认情况下不使用此文件。您可以更新 <component\_name>\_exdes.vhd 以代替 AXI 流量生成器）。 |
| clock\_gen.vhd | 用于示例设计的时钟生成模块。 |
| axi4\_write\_master.vhd | 读取路径数据生成器模块，用于示例设计。 |
| axis\_data\_read.vhd | 读取路径数据检查器模块以用于示例设计。 |
| axis\_write\_master.vhd | 写入路径数据生成器模块用于示例设计。 |
| axi\_s2mm\_read.vhd | 用于示例设计的写入路径数据检查器模块。 |
| axis\_ctrl\_read.vhd | MM2S CTRL 数据检查器 |
| axis\_sts\_master.vhd | S2MM STS 数据生成器 |
| sg\_mif.coe | 块内存用于存储BD的COE文件 |

表 5-2 显示了可用于运行仿真的测试台文件。

*表 5-2：* **模拟目录**

|  |  |
| --- | --- |
| **名字** | **描述** |
| <component\_name>\_exdes\_tb.vhd | 示例设计的测试台 |

表 5-3 显示了实现示例设计所需的 XDC 文件。

*表 5-3：* **约束目录**

|  |  |
| --- | --- |
| **名字** | **描述** |
| <component\_name>\_exdes.xdc | 示例设计的顶级约束文件。 |

示例设计随附的 XDC 具有为 KC705 板配置的所有 I/O 引脚。默认情况下，这些约束是注释的。在继续实施 KC705 板之前取消注释。

## 模拟示例设计

使用AXI D MA 示例设计（作为 AXI DMA 的一部分提供），可以快速模拟和观察 AXI DMA 的行为。

仿真脚本编译 AXI DMA 示例设计和支持仿真文件。然后，它运行模拟并检查以确保它成功完成。

如果测试失败，将显示以下消息： 测试失败!!!

如果测试通过，将显示以下消息： 测试成功完成

如果测试挂起，则显示以下消息： 测试失败！！Test Timed Out

## 示例设计的测试台

本节包含有关 Vivado 设计套件中提供的测试台的信息。

X-参考目标 - 图 5-2

时钟和重置

代

<

组件名称>\_ex

德斯.vhd

(

返回页首

)

测试状态

clock\_in

重置

top\_tb

做

地位

X13595

*图 5-2：* **AXI DMA 示例设计测试平台**

图 5-2 显示了 AXI DMA 示例设计的测试台。顶层测试台产生一个200 MHz差分时钟，并驱动示例设计的初始复位。



*附录 A*  升级

本附录包含有关将设计从ISE®迁移到Vivado®设计套件以及升级到较新版本的IP核的信息。对于在 Vivado 设计套件中进行升级的客户，包括有关任何端口更改和对用户逻辑的其他影响的重要详细信息（如果适用）。

## 迁移到 Vivado 设计套件

有关从ISE工具迁移到Vivado设计套件的信息，请参阅*ISE到Vivado设计套件迁移指南*（UG911）[参考文献8]。

## 在 Vivado 设计套件中进行升级

本节提供有关在 Vivado 设计套件中升级到此 IP 核的最新版本时对用户逻辑或端口名称所做的任何更改的信息。此内核没有参数或端口更改。



*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。

## 寻求有关 Xilinx.com 的帮助

为了在使用 AXI DMA 内核时帮助设计和调试过程，[Xilinx](https://www.xilinx.com/support)  支持[网页包含关键资源，例如产品文档、发行说明、答复记录、有关已知问题的信息以及用于获取更多产品支持的链接。](https://www.xilinx.com/support)

### 文档

本产品指南是与 AXI DMA 内核相关的主要文档。本指南以及与 有助于设计过程的所有产品相关的文档，可在 [Xilinx 支持网页 上或使用](https://www.xilinx.com/support) Xilinx® 文档导航器找到。

从[下载页面](https://www.xilinx.com/support/download.html)下载 Xilinx 文档导航器[。](https://www.xilinx.com/support/download.html) 有关此工具和可用功能的详细信息，请在安装后打开联机帮助。

### 答复记录

答复记录包括有关常见问题的信息、有关如何解决这些问题的帮助信息，以及 Xilinx 产品的任何已知问题。答复记录每天创建和维护，确保用户能够访问最准确的可用信息。

可以使用主上的“搜索支持”框查找此核心的答复记录

[赛灵思支持网页](https://www.xilinx.com/support)。要最大化搜索结果，请使用适当的关键字，例如

* 产品名称
* 工具消息
* 所遇到问题的摘要

返回结果后，可以使用筛选器搜索来进一步定位结果。

 *附录 B：* **调试**

**AXI DMA 内核的主**答复记录

赛灵思答复 [54682](https://www.xilinx.com/support/answers/54682.html)

### 技术支持

当按照产品文档中的说明使用时，[Xilinx 会在 Xilinx 支持网页](https://www.xilinx.com/support)中提供[此](https://www.xilinx.com/support) LogiCORE™ IP 产品的技术支持。如果您执行以下任一操作，Xilinx 无法保证时序、功能或支持：

* 在文档中未定义的设备中实现解决方案。
* 自定义超出产品文档中允许的解决方案。
* 更改标有“请勿修改”的设计的任何部分。

如需联系 Xilinx 技术支持，请导航至 [Xilinx 支持网页](https://www.xilinx.com/support)。

## Vivado 设计套件调试功能

Vivado® 设计套件调试功能可将逻辑分析仪和虚拟 I/O 内核直接插入到您的设计中。调试功能还允许您设置触发条件，以捕获硬件中的应用程序和集成块端口信号。然后可以分析捕获的信号。Vivado IDE 中的此功能用于对在 Xilinx 器件中运行的设计进行逻辑调试和验证。

Vivado 逻辑分析仪与逻辑调试 IP 内核配合使用，包括：

* ILA 2.0（及更高版本）
* VIO 2.0 （和 later 版本）

几个内部信号被标记为调试信号。这些可以很容易地添加到逻辑分析仪中。

请参见 *Vivado 设计套件用户指南：编程和调试* （UG908） [参考文献 9]。

*附录 B：* **调试**

## 硬件调试

硬件问题的范围从链路启动到数h测试后发现的问题。本节介绍常见问题的调试步骤。

以下是遇到的一些常见问题和可能的解决方案。

* 您已经对BD环进行了编程，但似乎没有任何效果。必须遵循寄存器编程顺序才能启动 DMA。请参阅编程顺序和描述符管理。
* 在状态寄存器中设置的内部错误/错误位

° 当描述符中指定的 BTT 为 0 时，将设置内部错误。

如果获取的BD是已完成的BD，则将设置°SG内部错误。

° 其他错误位，如解码错误或从错误，也将根据互连或从机的响应进行设置。

* 您正在从某个位置读取数据，但数据似乎没有 顺序。

验证起始地址位置是对齐还是未对齐。如果未对齐，请确保在配置 DMA 时启用 DRE。

另请参阅 [Xilinx 解决方案中心](https://www.xilinx.com/support/solcenters.htm)，[了解在设计周期各个阶段提供器件、软件工具和知识产权方面的支持。主题包括设计即服务、建议和故障排除提示。](https://www.xilinx.com/support/solcenters.htm)



*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)。

## 文档导航器和设计中心

Xilinx 文档导航器提供对 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 **Start > All Program > Xilinx Design Tools > 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 页面。](https://www.xilinx.com/cgi-bin/docs/ndoc?t=design+hubs)

***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

如需搜索 Xilinx 文档，请转至 [www.xilinx.com/support。](https://www.xilinx.com/support/)

*1.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) 2. *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)

1. [*AMBA AXI 和 ACE 协议规范* （](http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.ihi0022d/index.htm)[ARM IHI 0022D）](https://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.ihi0022d/index.html)
2. *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)
3. *Vivado 设计套件用户指南：使用 IP 集成器设计 IP 子系统* （[UG994）](https://www.xilinx.com/cgi-bin/docs/rdoc?v=latest;d=ug994-vivado-ip-subsystems.pdf)
4. *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)
5. *AXI 流量 LogiCORE IP 生成器* （[PG125）](https://www.xilinx.com/cgi-bin/docs/ipdoc?c=axi_traffic_gen;v=latest;d=pg125-axi-traffic-gen.pdf)
6. *ISE 到 Vivado 设计套件迁移指南* （[UG911）](https://www.xilinx.com/cgi-bin/docs/rdoc?v=latest;d=ug911-vivado-migration.pdf)
7. *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)
8. *AXI 互连 LogiCORE IP 产品指南* （[PG059）](https://www.xilinx.com/cgi-bin/docs/ipdoc?c=axi_interconnect;v=latest;d=pg059-axi-interconnect.pdf)
9. *综合和仿真设计指南* （[UG626）](https://www.xilinx.com/cgi-bin/docs/rdoc?v=latest+ise;d=sim.pdf)
10. *AXI 多通道直接内存访问* （[PG288）](https://www.xilinx.com/cgi-bin/docs/ipdoc?c=axi_mcdma;v=latest;d=pg288-axi-mcdma.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** |
| 04/27/2022 | 7.1 | * 更新了 “性能 ”部分 * 更新了 散点收集描述符 部分 |
| 06/14/2019 | 7.1 | * 更新了表 2-7 和表 2-17。 * 更新了图 2-25、图 4-1 和图 4-2。 * 添加了 启用单 AXI4 数据接口 部分。 |
| 04/04/2018 | 7.1 | 添加了对 64 MB 数据传输的支持。 |
| 10/04/2017 | 7.1 | * 向本附录添加了文档导航器和设计中心。 * 添加了汽车应用免责声明。 * 将数据重新对齐引擎支持更新为 512 位（为 64 位）。 |

|  |  |  |
| --- | --- | --- |
| **日期** | **版本** | **校订** |
| 10/05/2016 | 7.1 | * 在“寄存器空间”部分的开头添加了有关 AXI4-Lite 写访问寄存器的注释。 * “补充了。 * 更新了 S2MM 描述。 |
| 11/18/2015 | 7.1 | 添加了对 UltraScale+ 系列的支持。 |
| 04/01/2015 | 7.1 | * 修复了主答复记录的链接。 * 添加了对 64 位寻址的支持。 |
| 04/02/2014 | 7.1 | * 添加了有关空中交通流量生成器的信息。 * 添加了有关可选微型 DMA 的信息。 * 增加了对 I/O 信号的axi\_dma\_tstvec。 |
| 12/18/2013 | 7.1 | 添加了 UltraScale™ 体系结构支持。 |
| 10/02/2013 | 7.1 | * 添加了示例设计。 * 添加了循环 BD 启用。 * S2MM\_CONTROL寄存器的修改位 26 和 27。 * 更新了屏幕显示。 * 添加了 IP 集成商信息。 * 添加了启用微型 DMA 选项。 |
| 03/20/2013 | 7.0 | * 修订号提前到 7.0 以与核心版本号 7.0 保持一致。 * 针对 Vivado 设计套件支持和内核版本 7.0 进行了更新 * 更新了调试附录。 * 删除了第 4 章中的一个屏幕截图并更新了另一个屏幕截图。 * 移除了 ISE®、CORE Generator™、Virtex-6® 和 Spartan-6® 材料。 * 删除了第 3 章中的设计参数和 AXI DMA 系统配置部分。 |
| 12/18/2012 | 3.2 | * 针对 14.4/2012.4 支持和核心版本 6.03a 进行了更新。 * 更新了调试附录。 * 更新了第 4 章中的屏幕截图。 * 替换图 1-1。 * 表 2-1， 系统性能中更新的设备。 * 更新了表 2-4、2-5 和 2-6 中的资源编号。 * 删除了互连参数和允许的参数组合部分。 * 更新了第 4 章和第 7 章中的输出生成部分。 |
| 10/16/2012 | 3.1 | 针对 14.3/2012.3 支持进行了更新。  文档清理 |
| 07/25/2012 | 3.0 | 核心更改摘要  • 增加了 Vivado 工具支持和 Zynq-7000® 支持 |

|  |  |  |
| --- | --- | --- |
| **日期** | **版本** | **校订** |
| 04/24/2012 | 2.0 | 核心更改摘要   * 添加了多通道支持 * 添加了二维事务支持 * 添加了锁孔支持 * 为 AXI 内存端接口添加了缓存和用户控件 |
| 10/19/2011 | 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目的;（2） Xilinx 不对与材料（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负责（无论是合同还是侵权行为、包括疏忽或任何其他责任理论），包括任何直接、间接、特殊、包括、 或间接损失或损害（包括数据、利润、商誉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），即使此类损害或损失是可合理预见的，或者 Xilinx 已被告知可能发生此类损害或损失。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 Materials。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](https://www.xilinx.com/legal.htm" \l "tos) 查看[;](https://www.xilinx.com/legal.htm" \l "tos) IP 核可能受 Xi linx 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;您在此类关键应用中使用 Xilinx 产品承担全部风险和责任，请参阅 Xilinx 的销售条款，该条款可在 [https://www.xilinx.com/legal.htm#tos](https://www.xilinx.com/legal.htm" \l "tos) 查看[。](https://www.xilinx.com/legal.htm" \l "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”），除非有

符合ISO 26262 汽车安全标准（“安全”）的 SAF ETY 概念或冗余功能

设计“）。客户在使用或分发任何包含产品的系统之前，应出于安全目的对此类系统进行彻底测试。在没有安全设计的情况下在安全应用中使用PR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 2011–2022 Xilinx， Inc. Xilinx、Xilinx 徽标、Alveo、Artix、Kintex、Kria、Spartan、Versal、Vitis、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。所有其他商标均为其各自所有者的财产。