第3章

SRIO

RapidIO是一种通用、高带宽、低引脚数、基于数据包交换的系统级互连体系结构的一种开放式互连技术标准。它主要用作系统内部接口，用于以每秒千兆字节的性能水平进行芯片到芯片及板到板通信。KeyStone器件中使用的RapidIO外围设备称为串行RapidIO(Serial RapidIO，SRIO)。

SRIO接口支持4通道SRIO 2.1规范，每通道支持1.25/2.5/3.125/5Gb/s波特率传输。PCIE Gen2单个接口支持1通道或两通道，每通道最高可支持5Gb/s波特率。

SRIO端口在C6678器件上是一个高性能、接口数量少的内部连接方式。在一个基带板内部使用RapidIO进行连接设计，可以创建一个对等的互连环境，在部件间提供更多的连接和控制，性能可达GB/s。RapidIO基于处理器总线中存储器和器件寻址概念，事务处理完全由硬件管理。采用RapidIO互连不仅可以降低系统成本，还可以降低潜在风险，减小包数据处理的间接消耗并提供更高的系统带宽。

这种体系结构可以用于连接微处理器、内存和内存映射的I/O器件，通常应用于网络设备、内存子系统和通用处理系统。

RapidIO的主要特点包括：

(1) 灵活的系统架构，允许点对点通信。

(2) 具有错误检测功能、强大的通信能力。

(3) 频率和端口宽度可扩展性。

(4) 非软件密集型操作。

(5) 高带宽互连，低开销。

(6) 引脚数少。

(7) 低功率。

(8) 低延迟。

3.1 SRIO介绍

RapidIO体系架构和RapidIO互连结构分别如图3.1和图3.2所示。

SRIO中的RapidIO外部设备要求：SRIO为所有符合RapidIO物理层1×/4×LP系

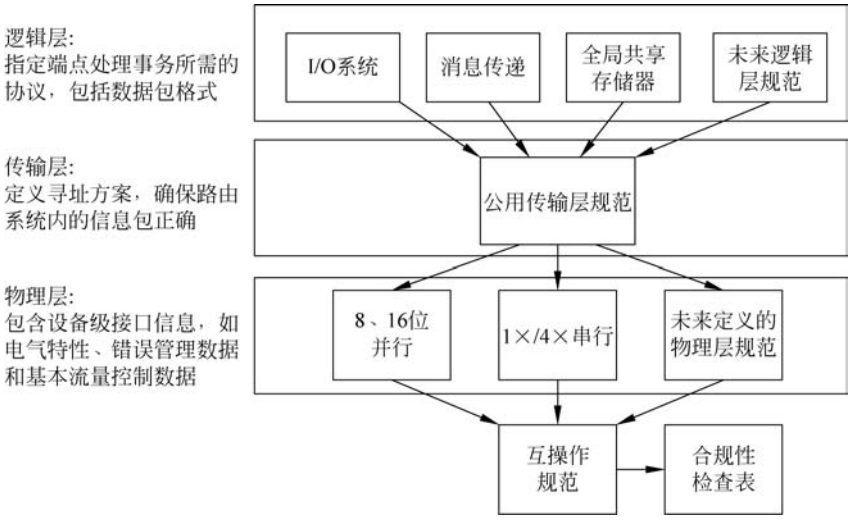

图 3.1 RapidIO 体系架构

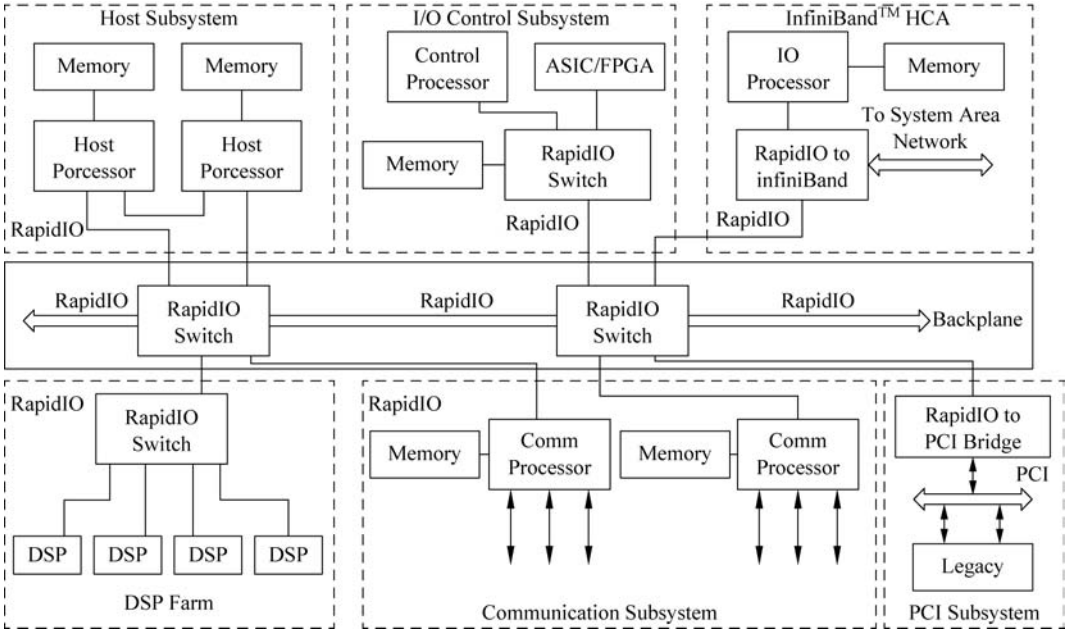

图 3.2 RapidIO 互连结构

列规范 V1.2 的设备提供无缝接口。这包括多个供应商提供的 ASIC、微处理器、DSP 和交叉开关网络(Switch Fabric)设备。

SRIO 的驱动库在 ti\pdk_C6678_x_x_x_x\packages\ti\drv\srio 目录下，相关文档在 docs 目录，例程在 example 目录下。

SRIO 的编程示例程序基于寄存器层 CSL(芯片支持层)。大多数代码使用如下定义的 SRIO 寄存器指针：

```
#include <ti/csl/cslr_srio.h>
```

```
#include <ti/csl/cslr_device.h>
CSL_SrioRegs * srioRegs = (CSL_SrioRegs *)CSL_SRIO_CONFIG_REGS;
```

3.1.1 物理层 1×/4×-LP 系列规范

目前，RapidIO 行业协会认可两种物理层规范：8/16 LP-LVDS 和 1×/4×LP 系列。8/16 LP-LVDS 规范是一个点对点同步时钟源 DDR 接口。1×/4×LP 串行规格是一个点对点、交流耦合、时钟恢复的接口。两个物理层规格不兼容。

SRIO 符合 1×/4×LP 系列规范。SRIO 中的 SerDes(Serializer/Deserializer)技术也与该规范一致。

RapidIO 物理层 1×/4×LP 串行规范目前涵盖四个频点：1.25Gb/s、2.5Gb/s、3.125Gb/s 和 5Gb/s。这定义了每个 I/O 信号差分对的总带宽。由于 8 位/10 位编码开销，每个差分对的有效数据带宽分别为 1.0Gb/s、2.0Gb/s、2.5Gb/s 和 4Gb/s。

SRIO 物理层结构如图 3.3 所示。

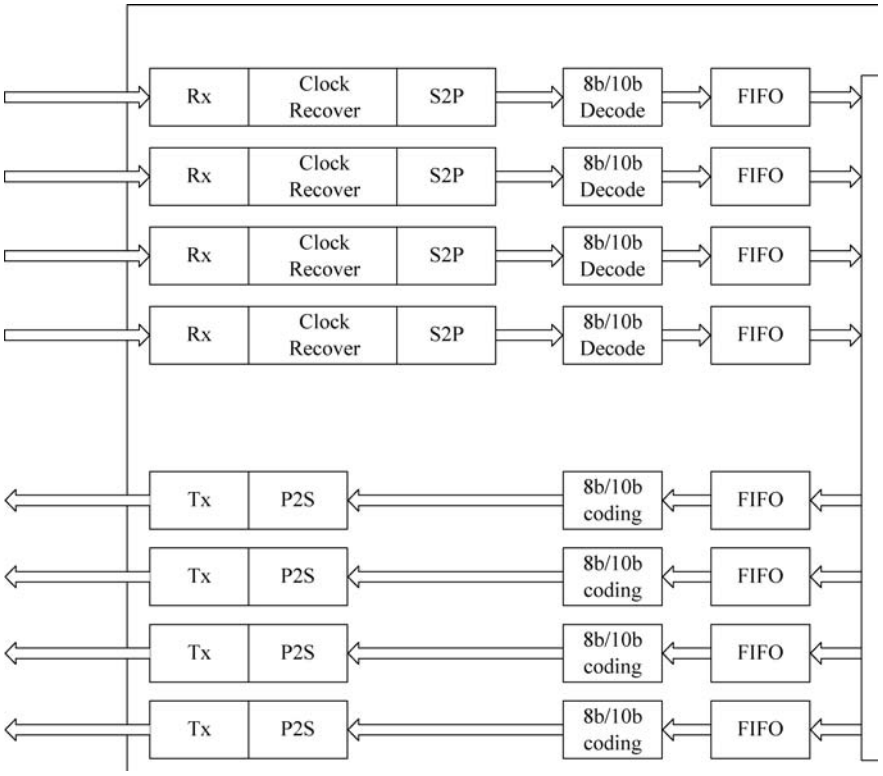

图 3.3 SRIO 物理层结构

1. 1×和 4×连接

SRIO 物理层 1×/4×互连结构如图 3.4 所示。SRIO 设备引脚采用高速差分信号接收数据。一个设备的每条正传输数据线(TDx)都连到另一个设备上对应编号的正接收数据线。同样，每条负传输数据线($\overline{\text{TDx}}$)连接到一个对应编号的负接收数据线($\overline{\text{RDx}}$)。图 3.4 中 RD 为接收数据线(Receive Data Line)的简写，TD 为传输数据线(Tranceive Data)的缩写。

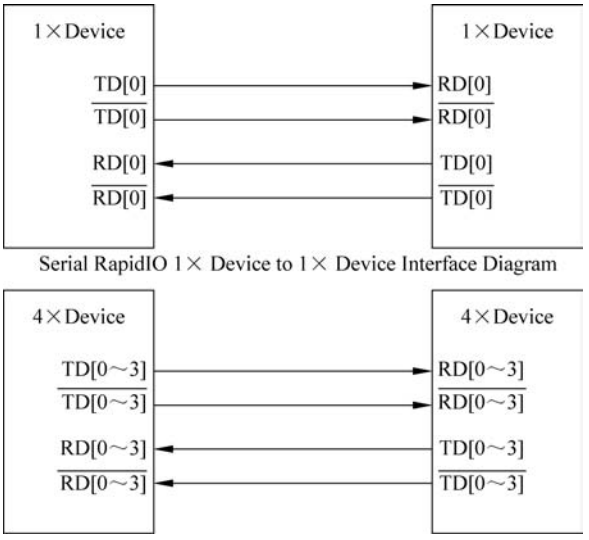

图 3.4 SRIO 1×和 4×互连结构

2. SRIO 引脚

SRIO 设备引脚是基于电流模式逻辑(Current-Mode Logic,CML)开关电平的高速差分信号。发送和接收缓冲区独立于时钟恢复块中。参考时钟输入未合并到 SerDes 宏中。它使用了一个差分输入缓冲器,与晶体振荡器制造商提供的 LVDS 和 LVPECL 接口兼容。表 3.1 描述了 SRIO 外围设备的引脚。

表 3.1 SRIO 外围设备的引脚

引 脚 名	引脚数	信号方向	描 述
RIOTX3/$\overline{\text{RIOTX3}}$	2	Output	发送数据-差分点对点单向总线。将数据包数据发送到接收设备的 Rx 引脚,用于 1 个 4×设备中的最高有效位,也可用于 4 个 1×设备
RIOTX2/$\overline{\text{RIOTX2}}$	2	Output	发送数据-差分点对点单向总线。将数据包数据发送到接收设备的 Rx 引脚,用于 4 个 1×设备和 1 个 4×设备的位
RIOTX1/$\overline{\text{RIOTX1}}$	2	Output	发送数据-差分点对点单向总线。将数据包数据发送到接收设备的 Rx 引脚,用于 4 个 1×设备和 1 个 4×设备的位
RIOTX0/$\overline{\text{RIOTX0}}$	2	Output	发送数据-差分点对点单向总线。将数据包数据发送到接收设备的 Rx 引脚,用于 1 个 1×设备、4 个 1×设备和 1 个 4×设备的位
RIORX3/$\overline{\text{RIORX3}}$	2	Input	接收数据-差分点对点单向总线。接收发送设备的 Tx 引脚的数据包数据。1 个 4×设备中的最高有效位,也可用于 4 个 1×设备
RIORX2/$\overline{\text{RIORX2}}$	2	Input	接收数据-差分点对点单向总线。接收发送设备的 Tx 引脚的数据包数据,用于 4 个 1×设备和 1 个 4×设备的位
RIORX1/$\overline{\text{RIORX1}}$	2	Input	接收数据-差分点对点单向总线。接收发送设备的 Tx 引脚的数据包数据,用于 4 个 1×设备和 1 个 4×设备的位
RIORX0/$\overline{\text{RIORX0}}$	2	Input	接收数据-差分点对点单向总线。接收发送设备 Rx 引脚的数据包数据。用于 1 个 1×设备、4 个 1×设备和 1 个 4×设备的位
RIOCLK/$\overline{\text{RIOCLK}}$	2	Input	外围时钟恢复电路的参考时钟输入缓冲器

3. RapidIO 支持的功能

(1) 符合 RapidIO 互连规范 REV2.1.1。

(2) 符合 LP 系列规范 REV2.1.1。

(3) 4×串行 RapidIO：

- 1×端口，可选操作(4)个 1×端口；
- 2×端口，可选操作(2)个 2×端口；
- 2×端口和 1×端口操作，可选(1)个 2×端口和(2)个 1×端口；
- 4×端口，(1)个 4×端口。

(4) TI SerDes 集成时钟恢复。

(5) 能够以不同波特率运行不同端口(仅支持整数倍速率，即支持 2.5Gb/s 和 5Gb/s，不支持 3.125Gb/s 和 5Gb/s)。

(6) 硬件错误处理，包括 CRC。

(7) 支持 1.25Gb/s、2.5Gb/s、3.125Gb/s 和 5Gb/s 的波特率。

(8) 未使用端口的电源关闭选项。

(9) Read、Write、Write w/response、Streaming Write、输出原子化(Out-going Atomic)、维护(Maintenance)操作。

(10) 中断生成到 CPU(门铃包和内部调度)。

(11) 支持 8 位和 16 位 DeviceID。

(12) 支持接收 34 位地址。

(13) 支持生成 34 位、50 位和 66 位地址。

(14) 支持数据大小：字节、半字、字、双字。

(15) 定义为大端(Big Endian)。

(16) DirectIO 传输。

(17) 消息传递传输。

(18) 数据有效负载达到 256 字节。

(19) 单个消息生成最多 16 个数据包。

(20) 弹性存储 FIFO，用于时钟域切换。

(21) 支持短线传输(Short Run：低功耗，板内或短的背板连接)和长线传输(Long Run：最低 50cm 和两个及以上连接器)，AC 驱动特性不一样。

(22) 支持错误管理扩展。

(23) 支持拥塞控制扩展。

(24) 支持组播(Multicast ID)。

(25) 支持短控制符号和长控制符号。

(26) 支持 IDLE1 和 IDLE2。

(27) 基于优先级和 CRF 的协议单元之间的严格优先级段交织。

不支持以下功能。

(1) 符合 Global Shared Memory Specification(GSM 规范)。

(2) 8/16 LP-LVDS 兼容。

(3) 目的地(Destination)支持 RapidIO 原子操作。

3.1.2 SRIO 外围数据流

该外围设备是一个外部驱动的从模块，能够作为 DSP 芯片内的主设备。这意味着外部设备可以根据需要将（突发写入）数据推送到 DSP，而不必向 CPU 生成中断或不依赖于 DSP EDMA。这有几个好处：减少了中断的总数，减少了与只读外围设备相关联的握手（延迟），并为其他任务释放了 EDMA。

SRIO 规定有效负载高达 256 字节的数据包。通常一个事务包含多个数据包。RapidIO 规定每条消息最多 16 个数据包。尽管为每个数据包事务生成了一个请求，以便 DMA 可以将数据传输到二级内存，但只有在消息的最后一个数据包之后才会生成中断。此中断通知 CPU 数据在二级内存中可用于处理。

作为一个端点设备，外围设备根据目标 ID 接收数据包。对于数据包接收，有两种模式可选项。第一个选项只接收 DestID 与本地 DeviceID 匹配的数据包，这提供了一个安全级别。第二个选项是系统组播操作。当通过表 3.2 组播操作的 DestID 检查启用组播时，所有与下面注释中提到的 DeviceID 匹配的传入数据包，表 3.3 中描述的组播 DeviceID 寄存器都被接收。

表 3.2 组播操作的 DestID 检查

模式（Mode）	操　作	log_tgt_id_dis（PER_SET_CNTL，Bit 27，0x0020）	tgt_id_dis（TLM_SP{0..3}_CONTROL Bit 21，0x1b380，0x1b400，0x1b480，0x1b500）	mtc_tgt_id_dis（TLM_SP{0..3}_CONTROL Bit 20，0x1b380，0x1b400，0x1b480，0x1b500）
A	主机只能使用 DESTID＝RIO_BASE_ID（0xB060）或其他 15 个 DeviceID 值之一执行维护枚举、发现和分配； 没有组播； 没有数据包转发	×	0	0
B	主机可以使用任何 DestID 执行维护枚举、发现和分配； 允许对 Base_ID 进行硬编码（引脚设置），然后主机读取维护数据； 没有组播； 没有数据包转发	×	0	1
C	主机可以执行维护枚举、发现，且仅使用 DestID ＝ RIO _ Base _ ID（0xB060）或其他 15 个 DeviceID 值之一进行分配； 支持 8 个本地组播组； 支持所有数据包类型的数据包转发	0	1	0

续表

模式（Mode）	操　作	log_tgt_id_dis（PER_SET_CNTL，Bit 27，0x0020）	tgt_id_dis（TLM_SP{0..3}_CONTROL Bit 21，0x1b380，0x1b400，0x1b480，0x1b500）	mtc_tgt_id_dis（TLM_SP{0..3}_CONTROL Bit 20，0x1b380，0x1b400，0x1b480，0x1b500）
D	主机可以使用任何 DestID 执行维护枚举、发现和分配； 允许对 Base_ID 进行硬编码（引脚设置），然后主机读取维护数据； 支持 8 个本地组播组； 不进行 FType 8 维护包的包转发	0	1	1
E	主机只能使用 DestID＝RIO_Base_ID（0xB060）或其他 15 个 DeviceID 值之一执行维护枚举、发现和分配； 支持无限组播/单播组； 没有数据包转发	1	1	0
F	主机可以使用任何 DestID 执行维护枚举、发现和分配； 允许对 Base_ID 进行硬编码（引脚设置），然后主机读取维护数据； 支持无限组播/单播组； 没有数据包转发	1	1	1

表 3.3　检查 DeviceID 的寄存器

DeviceID 类型	寄 存 器 名	偏 移 地 址
Multicast ID	RapidIO Multicast ID0	0x00C0
	RapidIO Multicast ID1	0x00C4
	RapidIO Multicast ID2	0x00C8
	RapidIO Multicast ID3	0x00CC
	RapidIO Multicast ID4	0x00D0
	RapidIO Multicast ID5	0x00D4
	RapidIO Multicast ID6	0x00D8
	RapidIO Multicast ID7	0x00DC

（1）其中 log_tgt_id_dis＝0b0，仅支持 DestID 等于 Base_ID 或任何其他 15 个允许的 DeviceID 或 8 个 MulticastID 的数据包；log_tgt_id_dis＝0b1，支持混杂（Promiscuous）ID/单播 ID。逻辑层接收所有数据包，而不考虑 DestID，并由相应的功能块处理。

（2）tgt_id_dis 和 mtc_tgt_id_dis 一起使用以配置混杂模式。tgt_id_dis 和 mtc_tgt_id_dis 组合的行为如表 3.4 所示。

表 3.4 tgt_id_dis 和 mtc_tgt_id_dis 组合的行为

端 点	TGT_ID_DIS	0	1	0	1
	MTC_TGT_ID_DIS	0	0	1	1
维护请求/保留数据包	匹配 Base DeviceID	路由到 LLM(Logical Layer Module)	路由到 LLM	路由到 LLM	路由到 LLM
	匹配 BRR(Base Routing Register)	由 BRR 路由	由 BRR 路由	路由到 LLM	路由到 LLM
	无匹配	忽略	路由到用户内核	路由到 LLM	路由到 LLM
除维护请求/保留数据包外的所有数据包	匹配 Base DeviceID	路由到用户内核	路由到用户内核	路由到用户内核	路由到用户内核
	匹配 BRR	路由到用户内核	路由到用户内核	路由到用户内核	路由到用户内核
	无匹配	忽略	路由到用户内核	忽略	路由到用户内核

下一步是时钟同步和数据对齐。这些功能由 FIFO 和线去偏斜(Lane De-skewing)块处理。FIFO 提供一种灵活存储机制,用于在恢复的时钟域和公共系统时钟之间转换时钟域。在 FIFO 之后,无论使用的是 1×、2×还是 4×模式,四个线都在频率和相位上同步。FIFO 是 8 字深。在 4×模式下,线去偏斜块用于对齐每个通道的字边界,从而使生成的 32 位字正确对齐。

CRC 错误检测块保持输入数据的运行计算,并且计算 1×、2×或 4×模式的预期 CRC 值。将期望值与接收数据包末尾的 CRC 值进行比较。

在数据包到达逻辑层后,对数据包字段进行解码并缓冲有效负载。根据接收数据包的类型控制 DMA 访问的功能块处理,图 3.5 描述了这些块。

加载/存储单元(Load/Store Unit,LSU)控制 DirectIO 数据包的传输,内存访问单元(Memory Access Unit,MAU)控制 DirectIO 数据包的接收。LSU 还控制维护包的传输。消息包由 TXU 发送并由 RXU 接收。这四个单元使用内部 DMA 与内部存储器通信,它们使用缓冲区和接收/发送端口与外部设备通信。

RapidIO 数据流由与逻辑层、传输层和物理层相关的数据字段组成。

(1) 逻辑层由头(定义访问类型)和有效负载(如果存在)组成。

(2) 传输层在某种程度上依赖于系统中的物理拓扑结构,由发送和接收设备的源和目标 ID 组成。

(3) 物理层依赖于物理接口(即串行与并行 RapidIO),包括优先级、确认和错误检查字段。

3.1.3 SRIO 包

SRIO 事务基于请求和响应数据包。包是系统中端点设备之间的通信元素。主机或发起者生成一个发送到目标的请求包。然后,目标生成一个响应包返回给发起方以完成事务。

SRIO 端点通常不直接连接到彼此,而是具有中间交换结构(Fabric)互连。如图 3.6 所

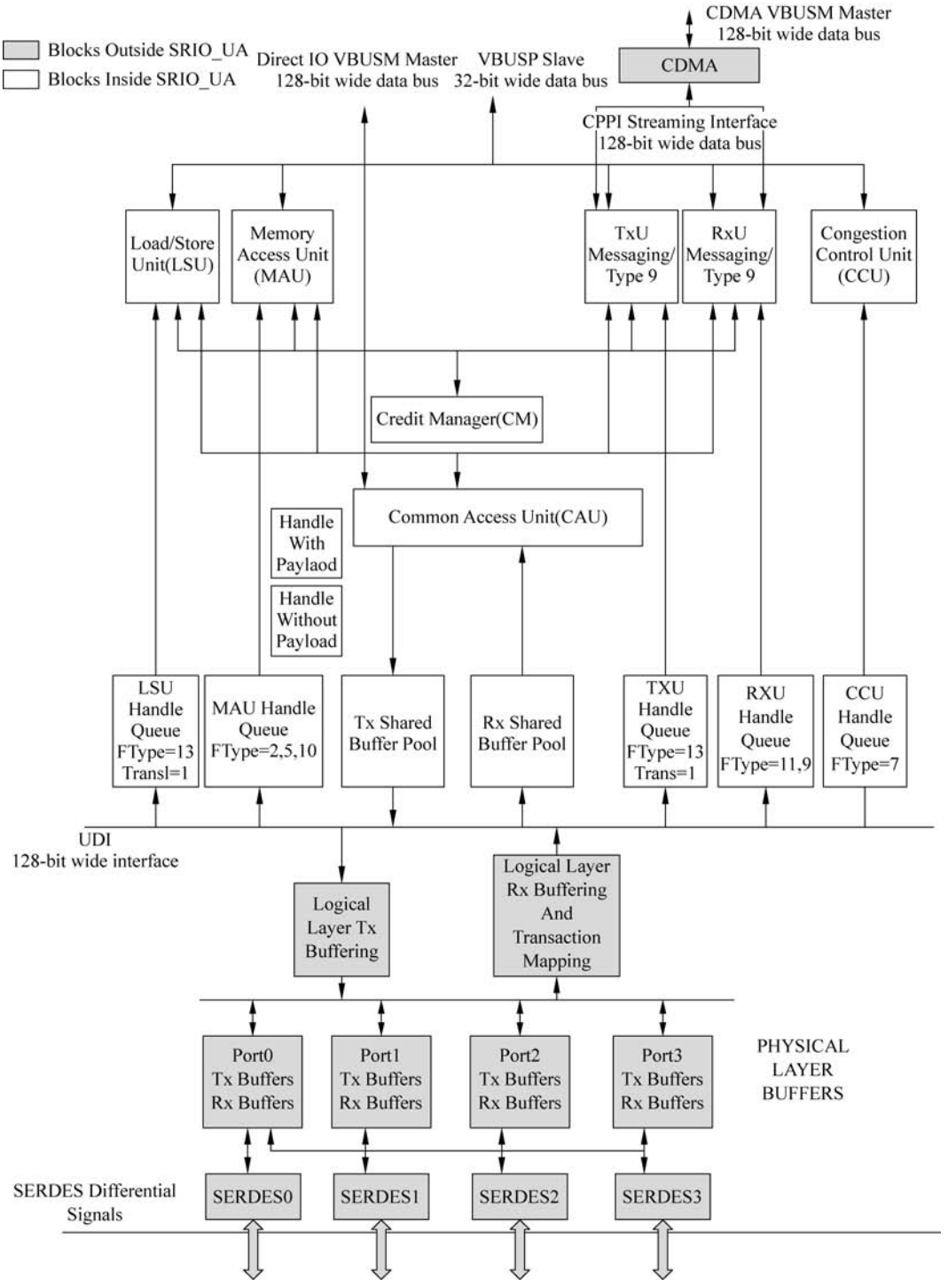

图 3.5 SRIO 外围模块

示为 SRIO 发起者与目标之间的操作顺序。控制符(Control Symbols)用于管理 SRIO 物理互连中的事务流。控制符用于包确认、流控制信息和维护功能。

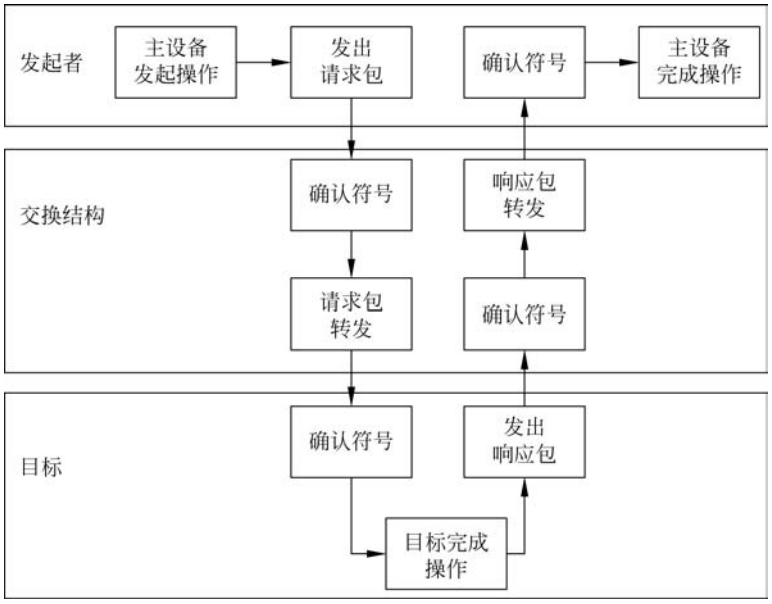

图 3.6 SRIO 操作顺序

1. 流式写(Streaming Write)包示例

图 3.7 描述了作为两个数据流的示例数据包。第一个适用于 80 字节或更小的有效负载,而第二个适用于 80 到 256 字节的有效负载。SRIO 数据包的长度必须是 32 位的偶数。如果物理层、逻辑层和传输层的组合长度为 16 位整数,则在 CRC 之后,在包的末尾添加一个值为 0000H 的 16 位 Pad(图中未描述)。定义为保留的 Reserved 位字段在生成时设置为 0,在接收时忽略。输入/输出逻辑规范(RapidIO Input/Output Logical Specification)和消息传递逻辑规范(Message Passing Logical Specification)中描述了所有请求和响应包格式。注:图 3.7 假设地址为 32 位,DeviceID 为 8 位。

DeviceID 如是一个 8 位字段,它最多可以寻址系统中的 256 个节点。如果使用 16 位地址,系统最多可以容纳 64K 个节点。

数据流包括一个循环冗余码(CRC)字段,以确保正确接收数据。CRC 值保护整个数据包,除了 ackID 和 PHY 字段的一位保留位(rsv)。外围设备在硬件上自动检查 CRC。如果 CRC 正确,接收设备将发送一个接受包的控制符。如果 CRC 不正确,则发送一个不接受的包控制符,以便重试传输。

2. 控制符

控制符(Control Symbols)是管理链路维护、包定界(Packet Delimiting)、包确认(Packet Acknowledgment)、错误报告和错误恢复的物理层消息元素。所有传输的数据包都由包的开始和包的结束分隔符分隔。SRIO 控制符的长度为 24 位,由它们自己的 CRC 保护(见图 3.8)。控制符号提供两种功能:stype0 符号表示发送该符号的端口的状态,stype1 符号是接收端口或发送分隔符(Transmission Delimiters)需要的。它们具有以下格式,详见 *Physical layer 1×/4×LP-Serial Specification* 的第 3 节。

控制符号由符号开头的特殊字符分隔。如果控制符号包含数据包分隔符(数据包开始、