RapidIO™ Interconnect Specification Part 3: Common Transport Specification

Rev. 1.3, 06/2005

RapidIO Trade Association

Revision History

Revision	Description	Date
1.1	First public release	03/08/2001
1.2	No technical changes	06/26/2002
1.3	Technical changes: the following new features showings: 03-01-00002.006, 03-03-00002.002 Converted to ISO-friendly templates; re-formatted	02/23/2005
1.3	Removed confidentiality markings for public release	06/07/2005

Questions regarding the RapidIO Trade Association, specifications, or membership should be forwarded to:

RapidIO Trade Association
Suite 325, 3925 W. Braker Lane
Austin, TX 78759
512-305-0070 Tel.
512-305-0009 FAX.

Table of Contents

Chapter 1 Overview

Chapter 2 Transport Format Description

Chapter 3 Common Transport Registers

Table of Contents

Table of Contents

Blank page

Table of Contents

List of Figures

List of Figures

List of Tables

List of Tables

Chapter 1 Overview

1.1 Introduction

This chapter provides an overview of the *RapidIO Part 3: Common Transport Specification*, including a description of the relationship between this specification and the other specifications of the RapidIO interconnect.

1.2 Overview

The *RapidIO Part 3: Common Transport Specification* defines a standard transport mechanism. In doing so, it specifies the header information added to a RapidIO logical packet and the way the header information is interpreted by a switching fabric. The RapidIO interconnect defines this mechanism independent of a physical implementation. The physical features of an implementation using RapidIO are defined by the requirements of the implementation, such as I/O signaling levels, interconnect topology, physical layer protocol, and error detection. These requirements are specified in the appropriate RapidIO physical layer specification.

This transport specification is also independent of any RapidIO logical layer specification.

1.3 Transport Layer Features

The transport layer functions of the RapidIO interconnect have been addressed by incorporating the following functional, physical, and performance features.

1.3.1 Functional Features

Functional features at the transport layer include the following:

- System sizes from very small to very large are supported in the same or compatible packet formats.
- Because RapidIO has only a single transport specification, compatibility among implementations is assured.
- The transport specification is flexible, so that it can be adapted to future applications.
- Packets are assumed, but not required, to be directed from a single source to a single destination.

1.3.2 Physical Features

The following are physical features of the RapidIO fabric that apply at the transport layer:

- The transport definition is independent of the width of the physical interface between devices in the interconnect fabric.
- No requirement exists in RapidIO for geographical addressing; a device's identifier does not depend on its location in the address map but can be assigned by other means.

1.3.3 Performance Features

Performance features that apply to the transport layer include the following:

- Packet headers are as small as possible to minimize the control overhead and are organized for fast, efficient assembly and disassembly.
- Broadcasting and multicasting can be implemented by interpreting the transport information in the interconnect fabric.
- Certain devices have bandwidth and latency requirements for proper operation. RapidIO does not preclude an implementation from imposing these constraints within the system.

1.4 Contents

RapidIO Part 3: Common Transport Specification contains three chapters:

- Chapter 1, "Overview" (this chapter) provides an overview of the specification
- Chapter 2, "Transport Format Description," describes the routing methods used in RapidIO for sending packets across the systems of switches described in this chapter.
- Chapter 3, "Common Transport Registers," describes the visible register set that allows an external processing element to determine the capabilities, configuration, and status of a processing element using this RapidIO transport layer definition.

1.5 Terminology

Refer to the Glossary at the back of this document.

1.6 Conventions

||	Concatenation, used to indicate that two fields are physically associated as consecutive bits
ACTIVE_HIGH	Names of active high signals are shown in uppercase text with

	no overbar. Active-high signals are asserted when high and not asserted when low.
$\overline{\text{ACTIVE\_LOW}}$	Names of active low signals are shown in uppercase text with an overbar. Active low signals are asserted when low and not asserted when high.
italics	Book titles in text are set in italics.
REG[FIELD]	Abbreviations or acronyms for registers are shown in uppercase text. Specific bits, fields, or ranges appear in brackets.
TRANSACTION	Transaction types are expressed in all caps.
operation	Device operation types are expressed in plain text.
n	A decimal value.
[*n*-*m*]	Used to express a numerical range from *n* to *m*.
0b*nn*	A binary value, the number of bits is determined by the number of digits.
0x*nn*	A hexadecimal value, the number of bits is determined by the number of digits or from the surrounding context; for example, 0x*nn* may be a 5, 6, 7, or 8 bit value.
x	This value is a don’t care

Blank page

Chapter 2 Transport Format Description

2.1 Introduction

This chapter contains the transport format definition for the *RapidIO Part 3: Common Transport Specification*. Three transport fields are added to the packet formats described in the RapidIO logical specifications. The transport formats are intended to be fabric independent so the system interconnect can be anything required for a particular application; therefore all descriptions of the transport fields and their relationship with the logical packets are shown as bit streams.

2.2 System Topology

RapidIO is intended to be interconnect fabric independent. This section describes several of the possible system topologies and routing methodologies allowed by the processing element models described in the Models chapters of the different Logical Specifications.

2.2.1 Switch-Based Systems

A RapidIO system can be organized around the concept of switches. Figure 2-1 shows a small system in which five processing elements are interconnected through two switches. A logical packet sent from one processing element to another is routed through the interconnect fabric by the switches by interpreting the transport fields. Because a request usually requires a response, the transport fields must somehow indicate the return path from the requestor to the responder.

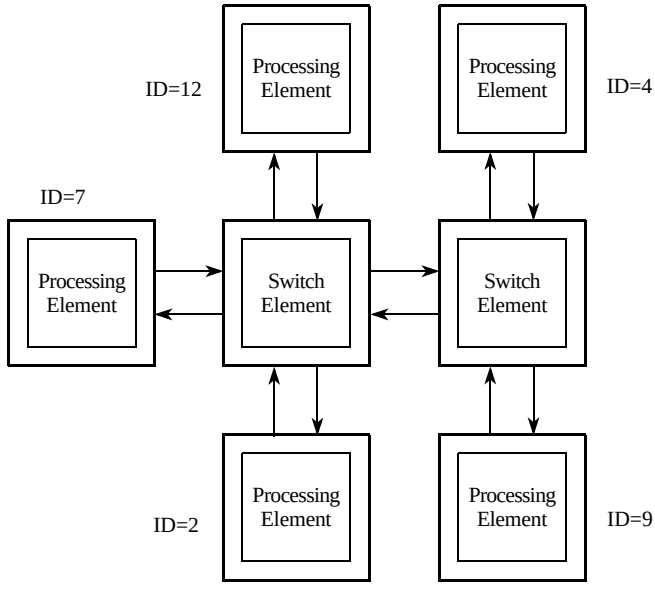

Figure 2-1. A Small Switch-Based System

2.2.2 Ring-Based Systems

A simplification of the switch structure is a ring as shown in Figure 2-2. A ring is a point-to-point version of a common bus; therefore, it is required to have a unique identifier for each processing element in the system. A packet put onto the ring contains the source and destination identifier in the transport fields. Each packet issued is examined by the downstream processing element. If that processing element's identifier matches that of the destination, it removes the packet from the ring for processing. If the destination identifier does not match the packet, it is passed to the next processing element in the ring.

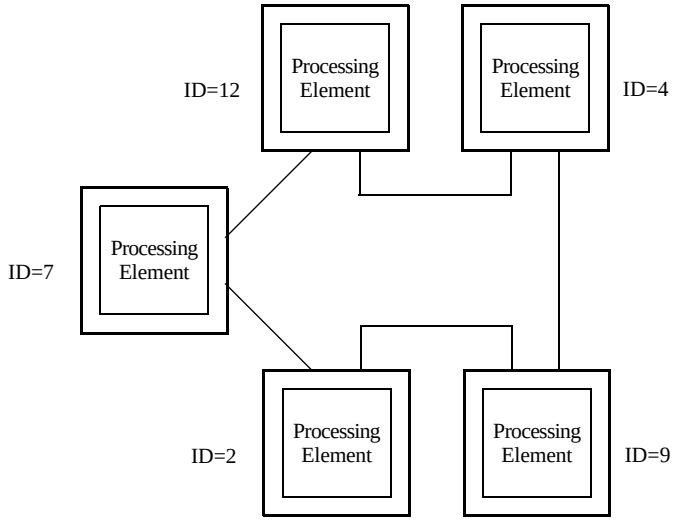

Figure 2-2. A Small Ring-Based System

2.3 System Packet Routing

There are many algorithms that can be used for routing through a system. The *RapidIO Part 3: Common Transport Specification* requires device identifier based packet routing. Each directly addressable device in the system shall have one or more unique device identifiers. When a packet is generated, the device ID of the destination of the packet is put in the packet header. The device ID of the source of the packet is also put in the packet header for use by the destination when generating response packets. When the destination of a request packet generates a response packet, it swaps the source and destination fields from the request, making the original source the new destination and itself the new source. Packets are routed through the fabric based on the destination device ID.

One method of routing packets in a switch fabric using device ID information incorporates routing tables. Each switch in the interconnect fabric contains a table that tells the switch how to route every destination ID from an input port to the proper output port. The simplest form of this method allows only a single path from every processing element to every other processing element. More complex forms of this method may allow adaptive routing for redundancy and congestion relief. However, the actual method by which packets are routed between the input of a switch and the output of a switch is implementation dependent.

2.4 Field Alignment and Definition

The *RapidIO Part 3: Common Transport Specification* adds a transport type (tt) field to the logical specification packet that allows four different transport packet types to be specified. The tt field indicates which type of additional transport fields are added

to the packet.

The three fields (tt, destinationID, and sourceID) added to the logical packets allow for two different sizes of the device ID fields, a large (16-bit), and a small (8-bit), as shown in Table 2-1. The two sizes of device ID fields allow two different system scalability points to optimize packet header overhead, and only affix additional transport field overhead if the additional addressing is required. The small device ID fields allow a maximum of 256 devices to be attached to the fabric. The large device ID fields allow systems with up to 65,536 devices.

Table 2-1. tt Field Definition

tt	Definition
0b00	8-bit deviceID fields
0b01	16-bit deviceID fields
0b10	Reserved
0b11	Reserved

Figure 2-3 shows the transport header definition bit stream. The shaded fields are the bits associated with the logical packet definition that are related to the transport bits. Specifically, the field labeled “Logical ftype” is the format type field defined in the logical specifications. This field comprises the first four bits of the logical packet. The second logical field shown (“Remainder of logical packet”) is the remainder of the logical packet of a size determined by the logical specifications, not including the logical ftype field which has already been included in the combined bit stream. The unshaded fields (tt=0b00 or tt=0b01 and destinationID and sourceID fields) are the transport fields added to the logical packet by the common transport specification.

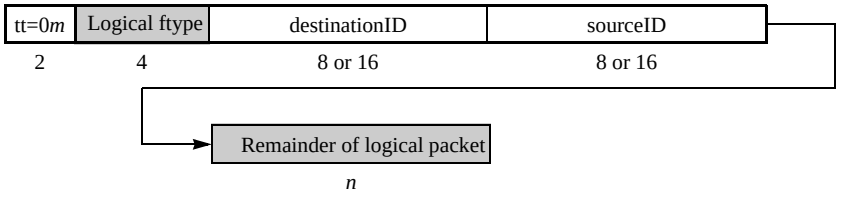

Figure 2-3. Destination-Source Transport Bit Stream

2.5 Routing Maintenance Packets

Routing maintenance packets in a switch-based network may be difficult because a switch processing element may not have its own device ID. An alternative method of addressing for maintenance packets for these devices uses an additional hop_count field in the packet to specify the number of switches (or hops) into the network from the issuing processing element that is being addressed. Whenever a switch processing element that does not have as associated device ID receives a maintenance packet it examines the hop_count field. If the received hop_count is

zero, the access is for that switch. If the hop_count is not zero, it is decremented and the packet is sent out of the switch according to the destinationID field. This method allows easy access to any intervening switches in the path between two addressable processing elements. However, since maintenance response packets are always targeted at an end point, the hop_count field shall always be assigned a value of 0xFF by the source of the packets to prevent them from being inadvertently accepted by an intervening device. Figure 2-4 shows the transport layer fields added to a maintenance logical packet. Maintenance logical packets can be found in the *RapidIO Part 1: Input/Output Logical Specification.*

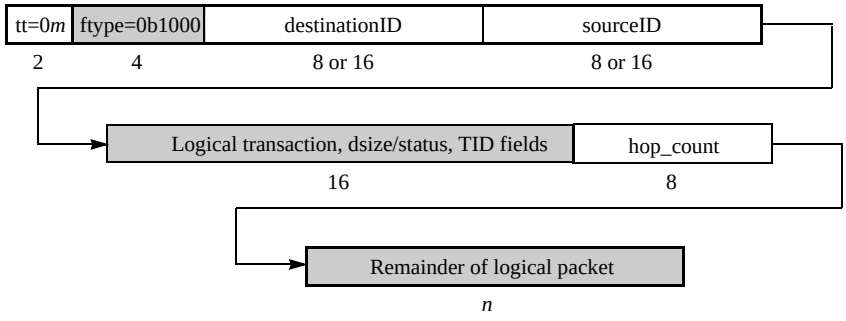

Figure 2-4. Maintenance Packet Transport Bit Stream

Blank page

Chapter 3 Common Transport Registers

3.1 Introduction

This chapter describes the visible register set that allows an external processing element to determine the capabilities, configuration, and status of a processing element using this transport layer definition. This chapter only describes registers or register bits defined by this specification. Refer to the other RapidIO logical, transport, and physical specifications of interest to determine a complete list of registers and bit definitions. All registers are 32-bits and aligned to a 32-bit boundary.

3.2 Register Summary

Table 3-1 shows the register address map for this RapidIO specification. These capability registers (CARs) and command and status registers (CSRs) can be accessed using *RapidIO Part 1: Input/Output Logical Specification* maintenance operations. Any register offsets not defined are considered reserved for this specification unless otherwise stated. Other registers required for a processing element are defined in other applicable RapidIO specifications and by the requirements of the specific device and are beyond the scope of this specification. Read and write accesses to reserved register offsets shall terminate normally and not cause an error condition in the target device. Writes to CAR (read-only) space shall terminate normally and not cause an error condition in the target device.

Register bits defined as reserved are considered reserved for this specification only. Bits that are reserved in this specification may be defined in another RapidIO specification.

Table 3-1. Common Transport Register Map

Configuration Space Byte Offset	Register Name
0x0-C	Reserved
0x10	Processing Element Features CAR
0x14–30	Reserved
0x34	Switch Route Table Destination ID Limit CAR
0x38-5C	Reserved

Table 3-1. Common Transport Register Map (Continued)

Configuration Space Byte Offset	Register Name
0x60	Base Device ID CSR
0x64	Reserved
0x68	Host Base Device ID Lock CSR
0x6C	Component Tag CSR
0x70	Standard Route Configuration Destination ID Select CSR
0x74	Standard Route Configuration Port Select CSR
0x78	Standard Route Default Port CSR
0x7C–FC	Reserved
0x100–FFFC	Extended Features Space
0x10000–FFFFFC	Implementation-defined Space

3.3 Reserved Register and Bit Behavior

Table 3-2 describes the required behavior for accesses to reserved register bits and reserved registers for the RapidIO register space,

Table 3-2. Configuration Space Reserved Access Behavior

Byte Offset	Space Name	Item	Initiator behavior	Target behavior
0x0–3C	Capability Register Space (CAR Space - this space is read-only)	Reserved bit	read - ignore returned value[1]	read - return logic 0
			write -	write - ignored
		Implementation-defined bit	read - ignore returned value unless implementation-defined function understood	read - return implementation-defined value
			write -	write - ignored
		Reserved register	read - ignore returned value	read - return logic 0s
			write -	write - ignored

Table 3-2. Configuration Space Reserved Access Behavior (Continued)

Byte Offset	Space Name	Item	Initiator behavior	Target behavior
0x40–FC	Command and Status Register Space (CSR Space)	Reserved bit	read - ignore returned value	read - return logic 0
			write - preserve current value[2]	write - ignored
		Implementation-defined bit	read - ignore returned value unless implementation-defined function understood	read - return implementation-defined value
			write - preserve current value if implementation-defined function not understood	write - implementation-defined
		Reserved register	read - ignore returned value	read - return logic 0s
			write -	write - ignored
0x100–FFFC	Extended Features Space	Reserved bit	read - ignore returned value	read - return logic 0
			write - preserve current value	write - ignored
		Implementation-defined bit	read - ignore returned value unless implementation-defined function understood	read - return implementation-defined value
			write - preserve current value if implementation-defined function not understood	write - implementation-defined
		Reserved register	read - ignore returned value	read - return logic 0s
			write -	write - ignored
0x10000–FFFFFC	Implementation-defined Space	Reserved bit and register	All behavior implementation-defined	

[1]Do not depend on reserved bits being a particular value; use appropriate masks to extract defined bits from the read value.

[2]All register writes shall be in the form: read the register to obtain the values of all reserved bits, merge in the desired values for defined bits to be modified, and write the register, thus preserving the value of all reserved bits.

3.4 Capability Registers (CARs)

Every processing element shall contain a set of registers that allows an external processing element to determine its capabilities using the I/O logical maintenance read operation. All registers are 32 bits wide and are organized and accessed in 32-bit (4 byte) quantities, although some processing elements may optionally allow larger accesses. CARs are read-only. Refer to Table 3-2 for the required behavior for accesses to reserved registers and register bits.

CARs are big-endian with bit 0 the most significant bit.

3.4.1 Processing Element Features CAR (Configuration Space Offset 0x10)

The processing element features CAR identifies the major functionality provided by the processing element. The bit settings are shown in Table 3-3.

Table 3-3. Bit Settings for Processing Element Features CAR

Bits	Name	Description
0–21	—	Reserved
22	Extended route table configuration support	0b0 - Switch PE does not support the extended route table configuration mechanism 0b1 - Switch PE supports the extended route table configuration mechanism (can only be set if bit 23 is set)
23	Standard route table configuration support	0b0 - Switch PE does not support the standard route table configuration mechanism 0b1 - Switch PE supports the standard route table configuration mechanism
22–26	—	Reserved
27	Common transport large system support	0b0 - PE does not support common transport large systems 0b1 - PE supports common transport large systems
28–31	—	Reserved

3.4.2 Switch Route Table Destination ID Limit CAR (Configuration Space Offset 0x34)

The Switch Route Table Destination ID Limit CAR specifies the maximum destination ID value that can be programmed with the standard route table configuration mechanism, and thereby indirectly defining the size of the route table. A route table access or extended route table access attempt to destination IDs greater than that specified in this register will have undefined results. This register is required if bit 23 of the Processing Element Features CAR is set. The bit settings are shown in Table 3-4.

Table 3-4. Bit Settings for Switch Route Table Destination ID Limit CAR

Bits	Name	Description
0–15	—	Reserved
16–31	Max_destID	Maximum configurable destination ID 0x00 - 1 destination ID 0x01 - 2 destinations IDs 0x02 - 3 destination IDs ... 0xFF - 65536 destination IDs

3.5 Command and Status Registers (CSRs)

A processing element shall contain a set of registers that allows an external processing element to control and determine status of its internal hardware. All registers are 32 bits wide and are organized and accessed in the same way as the CARs. Refer to Table 3-2 for the required behavior for accesses to reserved registers and register bits.

3.5.1 Base Device ID CSR (Configuration Space Offset 0x60)

The base device ID CSR contains the base device ID values for the processing element. A device may have multiple device ID values, but these are not defined in a standard CSR. The bit settings are shown in Table 3-5.

Table 3-5. Bit Settings for Base Device ID CSR

Bits	Name	Reset Value	Description
0-7	—		Reserved
8-15	Base_deviceID	see footnote[1]	This is the base ID of the device in a small common transport system (end point devices only)
16–31	Large_base_deviceID	see footnote[2]	This is the base ID of the device in a large common transport system (only valid for end point device and if bit 27 of the Processing Element Features CAR is set)

[1]The Base_deviceID reset value is implementation dependent

[2]The Large_base_deviceID reset value is implementation dependent

3.5.2 Host Base Device ID Lock CSR (Configuration Space Offset 0x68)

The host base device ID lock CSR contains the base device ID value for the processing element in the system that is responsible for initializing this processing element. The Host_base_deviceID field is a write-once/reset-able field which provides a lock function. Once the Host_base_deviceID field is written, all subsequent writes to the field are ignored, except in the case that the value written matches the value contained in the field. In this case, the register is re-initialized to 0xFFFF. After writing the Host_base_deviceID field a processing element must then read the Host Base Device ID Lock CSR to verify that it owns the lock before attempting to initialize this processing element. The bit settings are shown in Table 3-6.

Table 3-6. Bit Settings for Host Base Device ID Lock CSR

Bits	Name	Reset Value	Description
0-15	—		Reserved
16–31	Host_base_deviceID	0xFFFF	This is the base device ID for the PE that is initializing this PE.

3.5.3 Component Tag CSR (Configuration Space Offset 0x6C)

The component tag CSR contains a component tag value for the processing element and can be assigned by software when the device is initialized. It is especially useful for labeling and identifying devices that are not end points and do not have device ID registers. The bit settings are shown in Table 3-7.

Table 3-7. Bit Settings for Component ID CSR

Bits	Name	Reset Value	Description
0–31	component_tag	All 0s	This is a component tag for the PE.

3.5.4 Standard Route Configuration Destination ID Select CSR (Configuration Space Offset 0x70)

The Standard Route Configuration Destination ID Select CSR specifies the destination ID entry in the switch routing table to access when the Standard Route Configuration Port Select CSR is read or written.

The Ext_config_en bit controls whether the extended route table configuration mechanism is enabled. If the extended route table configuration mechanism is enabled, the specified destination ID and the next three sequential destination IDs are written or read when the Standard Route Configuration Port Select CSR accessed. Extended accesses that increment past the maximum specifiable destination ID (for example, starting an extended access at device ID 0xFF in a small transport system), has undefined results.

This register is required if bit 23 of the Processing Element Features CAR is set. The bit settings are shown in Table 3-8.

Table 3-8. Bit Settings for Standard Route Configuration Destination ID Select CSR

Bits	Name	Reset Value	Description
0	Ext_config_en	0b0	Extended Configuration Enable 0b0 - Extended configuration support is disabled 0b1 - Extended configuration support is enabled (only valid if bit 22 of the Processing Element Features CAR is set)
1-15	—		Reserved

Table 3-8. Bit Settings for Standard Route Configuration Destination ID Select CSR (Continued)

Bits	Name	Reset Value	Description
16-23	Config_destID_msb	0x00	Configuration destination ID most significant byte (only valid if bit 27 of the Processing Element Features CAR is set and the processing element is configured to operate in large transport mode)
24-31	Config_destID	0x00	Configuration destination ID

3.5.5 Standard Route Configuration Port Select CSR (Configuration Space Offset 0x74)

When written, the Standard Route Configuration Port Select CSR updates the switch output port configuration for packets with the destination ID selected by the Standard Route Configuration Destination ID Select CSR. When read, the Standard Route Configuration Port Select CSR returns the switch output port configuration for packets with the destination ID selected by the Standard Route Configuration Destination ID Select CSR.

If the extended route table configuration mechanism is enabled, when the Standard Route Configuration Port Select register is written the following route table configurations are carried out:

- destination ID Config_destID is routed to output port Config_output_port
- destination ID Config_destID+1 is routed to output port Config_output_port1
- destination ID Config_destID+2 is routed to output port Config_output_port2
- destination ID Config_destID+3 is routed to output port Config_output_port3

For reads of the Standard Route Configuration Port Select CSR, the configuration information is returned in the corresponding fashion.

After complete system initialization the switch output port route configuration information read may not be consistant with previously read values due to the capabilities and features of the particular switch. This register is required if bit 23 of the Processing Element Features CAR is set. The bit settings are shown in Table 3-9.

Table 3-9. Bit Settings for Standard Route Configuration Destination ID Select CSR

Bits	Name	Reset Value	Description
0-7	Config_output_port3	0x00	Configuration output port3 - This field is reserved if extended route table mechanism is not enabled
8-15	Config_output_port2	0x00	Configuration output port2 - This field is reserved if extended route table mechanism is not enabled
16-23	Config_output_port1	0x00	Configuration output port1 - This field is reserved if extended route table mechanism is not enabled
24-31	Config_output_port	see footnote[1]	Configuration output port

[1]The Config_output_port*n* reset values are implementation dependent

3.5.6 Standard Route Default Port CSR (Configuration Space Offset 0x78)

The Standard Route Default Port CSR specifies the port to which packets with destinations IDs that are greater than that specified in the Switch Route Table Destination ID Limit CAR are routed. This register is required if bit 23 of the Processing Element Features CAR is set. The bit settings are shown in Table 3-10.

Table 3-10. Bit Settings for Standard Route Default Port CSR

Bits	Name	Reset Value	Description
0-23	—		Reserved
24–31	Default_output_port	0x00	Default output port

Glossary of Terms and Abbreviations

The glossary contains an alphabetical list of terms, phrases, and abbreviations used in this book.

B

Big-endian. A byte-ordering method in memory where the address n of a word corresponds to the most significant byte. In an addressed memory word, the bytes are ordered (left to right) 0, 1, 2, 3, with 0 being the most significant byte.

Broadcast. The concept of sending a packet to all processing elements in a system.

C

Capability registers (CARs). A set of read-only registers that allows a processing element to determine another processing element's capabilities.

Command and status registers (CSRs). A set of registers that allows a processing element to control and determine the status of another processing element's internal hardware.

D

Destination. The termination point of a packet on the RapidIO interconnect, also referred to as a target.

Device. A generic participant on the RapidIO interconnect that sends or receives RapidIO transactions, also called a processing element.

Device ID. The identifier of an end point processing element connected to the RapidIO interconnect.

E

End point. A processing element which is the source or destination of transactions through a RapidIO fabric.

End point device. A processing element which contains end point functionality.

External processing element. A processing element other than the processing element in question.

F

Field or Field name. A sub-unit of a register, where bits in the register are named and defined.

H

Host. A processing element responsible for exploring and initializing all or a portion of a RapidIO based system.

I

Initiator. The origin of a packet on the RapidIO interconnect, also referred to as a source.

I/O. Input-output.

M

MSB. Most significant byte.

Multicast. The concept of sending a packet to more than one processing elements in a system.

O

Operation. A set of transactions between end point devices in a RapidIO system (requests and associated responses) such as a read or a write.

P

Packet. A set of information transmitted between devices in a RapidIO system.

Processing Element (**PE**). A generic participant on the RapidIO interconnect that sends or receives RapidIO transactions, also called a device.

S

Source. The origin of a packet on the RapidIO interconnect, also referred to as an initiator.

Switch. A multiple port processing element that directs a packet received on one of its input ports to one of its output ports.

T

Target. The termination point of a packet on the RapidIO interconnect, also referred to as a destination.

Transaction. A specific request or response packet transmitted between end point devices in a RapidIO system.

Blank page

Blank page