RapidIO™ Interconnect Specification Part 3: Common Transport Specification

4.1, 6/2017

RapidIO.org

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
2.0	Technical changes: errata showing 06-02-00001.005	06/14/2007
2.1	No technical changes	07/09/2009
2.1	Removed confidentiality markings for public release	08/13/2009
2.2	Technical changes: errata showings 10-08-00000.003, 10-08-00001.005, Consolidated Comments on 11-01-00000.000	05/05/2011
3.0	Changed RTA contact information. Technical changes: Support for 32-bit device IDs in packet formats. Added register block for Dev32 standard routing table support. Added Annex A Dev32 Programming Model Examples (Informative) Changed RTA contact information. Added Implementation Specific field to “Standard Route Default Port CSR”	10/11/2013
3.1	Minor typographical errors corrected. Technical changes: Added implementation specific bits to “Standard Route Cfg Port Select CSR”.	09/18/2014
3.2	Added resolution of Errata 13 Dev32 Routing Table Examples Correction.	01/28/2016
4.0	No technical changes.	06/15/2016
4.1	No technical changes.	06/30/2017

Questions regarding RapidIO.org, specifications, or membership should be forwarded to:

RapidIO.org
8650 Spicewood Springs #145-515
Austin, TX 78759
512-827-7680 Tel.

Table of Contents

Chapter 1 Overview

Chapter 2 Transport Format Description

Chapter 3 Common Transport Registers

Table of Contents

List of Figures

List of Figures

Blank page

List of Tables

List of Tables

Blank page

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.

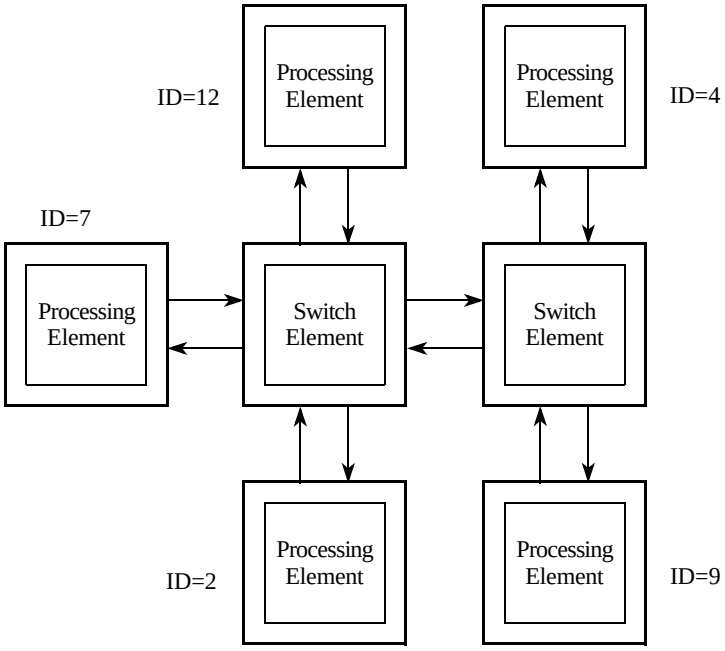

Figure 2-1. A Small Switch-Based System

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.

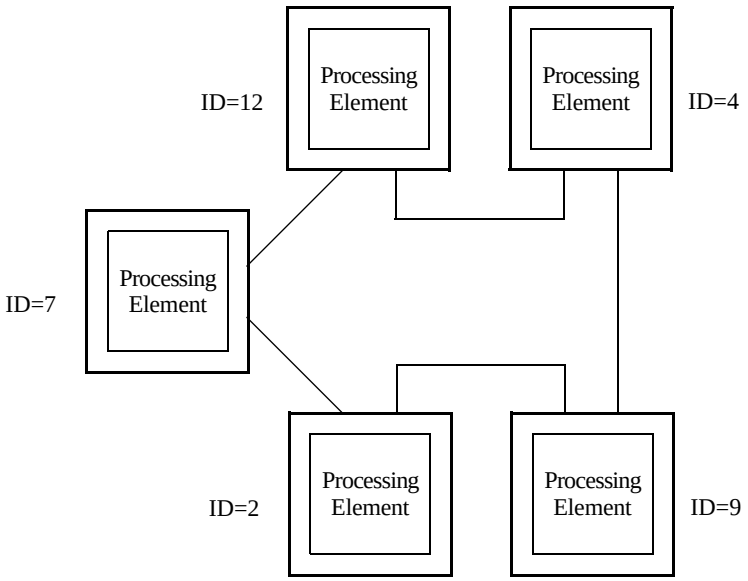

Figure 2-2. A Small Ring-Based System

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. All devices must provide a mechanism such that the receiver of a request shall process any received supported request regardless of device ID values. 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. Many applications and system designers may require that devices also provide some means to restrict processing requests to a list of acceptable device IDs. For those applications, it is recommended, however, that a device process all maintenance read requests. The behavior when a request is received with a device ID that is not on the acceptable list is implementation-dependant. 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 three different sizes of the device ID fields: Dev32 (32-bit), Dev16 (16-bit), and a Dev8 (8-bit), as shown in Table 2-1. The three sizes of device ID fields allow three different system scalability points to optimize packet header overhead, and only affix additional transport field overhead if the additional addressing is required. The Dev32 fields enable large system scalability while allowing packets to be routed to specific functions within a device based on destination ID. The Dev8 fields allow a maximum of 256 devices to be attached to the fabric. The Dev16 fields allow systems with up to 65,536 devices. The Dev32 fields allow systems with up to 4,294,967,296 devices.

Table 2-1. tt Field Definition

tt	Definition
0b00	Dev8 8-bit deviceID fields
0b01	Dev16 16-bit deviceID fields
0b10	Dev32 32-bit deviceID fields
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, destinationID and sourceID fields) are the transport fields added to the logical packet by the common transport specification.

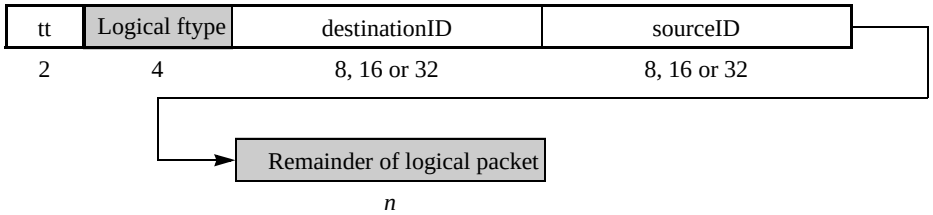

Figure 2-3. Destination-Source Transport Bit Stream

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. A switch processing element shall decrement the hop count when it forwards a maintenance packet. 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*.

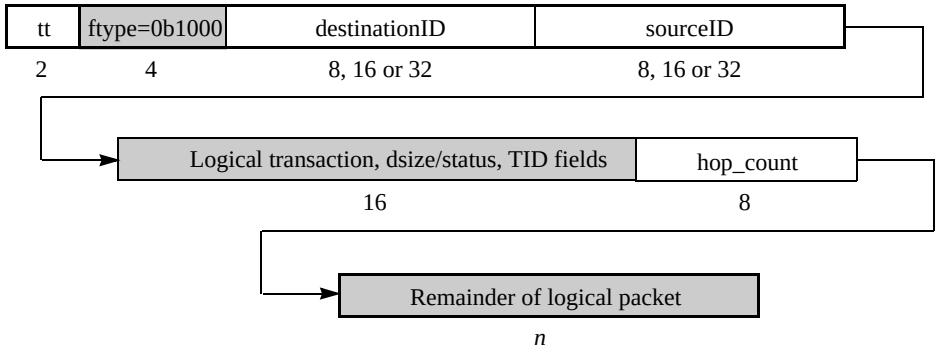

Figure 2-4. Maintenance Packet Transport Bit Stream

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	Dev32 Base Device ID CSR
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, Bit and Bit Field Value 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.

When a writable bit field is set to a reserved value, device behavior is implementation specific.

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
0–18	—	Reserved
19	Dev32 Support	0b0 - PE does not support Common Transport Dev32 0b1 - PE supports Common Transport Dev32
20-21	—	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 and bit 19 is clear)
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
24–26	—	Reserved
27	Dev16 support	0b0 - PE does not support Dev16 0b1 - PE supports Dev16
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 shall be implemented 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 0x0000 - 1 destination ID 0x0001 - 2 destination IDs 0x0002 - 3 destination IDs ... 0xFFFF - 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 Dev8 and Dev16 base device ID values for the processing element. A device can 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	Dev8_Base_deviceID	see footnote[1]	This is the Dev8 device ID of the device (endpoint devices only)
16–31	Dev16_base_deviceID	see footnote[2]	This is the Dev16 device ID of the device (must be valid for endpoint devices when bit 27 of the Processing Element Features CAR is set)

[1]The Dev8_Base_deviceID reset value is implementation dependent

[2]The Dev16_base_deviceID reset value is implementation dependent

3.5.2 Dev32 Base Device ID CSR (Configuration Space Offset 0x64)

The Dev32 base device ID CSR contains the Dev32 base device ID value for the processing element. A device can have multiple device ID values but these are not defined in a standard CSR. The bit settings are shown in Table 3-6.

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

Bits	Name	Reset Value	Description
0-31	Dev32_Base_DeviceID	see footnote[1]	This is the Dev32 device ID of the device (must be valid for endpoint devices when bit 19 of the Processing Element Features CAR is set)

[1]The Dev32_Base_DeviceID reset value is implementation dependent

3.5.3 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 Device ID Lock CSR is a write-once/reset-able register which provides a lock function. Once the Host Base Device ID Lock CSR is written, all subsequent writes to the register are ignored, except in the case that the value written matches the value contained in the register. In this case, the register is re-initialized to 0x0000_FFFF. After writing the Host Base Device ID Lock CSR 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 3-7.

Table 3-7. Bit Settings for Host Base Device ID Lock CSR

Bits	Name	Reset Value	Description
0-15	Host_base_Dev32ID	0x0000	If the Processing Element Features CAR Dev32 Support bit is 0, then this field is Reserved and shall have a constant value of 0. If the Processing Element Features CAR Dev32 Support bit is 1, this field contains the most significant 16 bits of the Dev32 base device ID for the PE that is initializing this PE.
16–31	Host_base_deviceID	0xFFFF	This is the base device ID for the PE that is initializing this PE.

3.5.4 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 3-8.

Table 3-8. 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.

3.5.5 Standard Route Cfg 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 is accessed. Extended accesses that increment past the maximum specifiable destination ID (for example, starting an extended access at device ID 0xFF in a Dev8 transport system) 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 3-9.

Table 3-9. 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
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 Dev16 transport mode)
24-31	Config_destID	0x00	Configuration destination ID

3.5.6 Standard Route Cfg 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 consistent with previously read values due to the capabilities and features of the particular switch. This register shall be implemented 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 Configuration Port Select CSR

Bits	Name	Reset Value	Description
0-3	Cop3_msb_or_imp_spec	0x00	Cop3_msb_or_imp_spec: This field shall be reserved if extended route table mechanism is not enabled and bit 19 of the Processing Element Features CAR is clear. When bit 19 of the Processing Element Features CAR is clear, and the extended route table mechanism is enabled, this field contains the most significant 4 bits of the configuration output port3 value. When bit 19 of the Processing Element Features CAR is set, this field optionally controls implementation-specific routing functionality: • Bits in this field that do not control implementation specific routing functionality shall be read only, with a fixed value of 0. • Implementation-specific routing functionality may be active if any bit in this field is set. • Implementation-specific routing functionality shall not be active if all bits in this field are clear.
4-7	Config_output_port3_lsb	0x00	Configuration output port3 - This field shall be reserved if the extended route table mechanism is not enabled. If the extended route table mechanism is enabled, this field contains the least significant 4 bits of the config output port3 value.
8-15	Config_output_port2	0x00	Configuration output port2 - This field shall be reserved if extended route table mechanism is not enabled
16-21	Config_output_port1_msb	0x00	Most significant 6 bits of the output port 1 value if the extended route table mechanism is enabled. This field shall be reserved if extended route table mechanism is not enabled.
22-23	Config_output_port1_lsb	0b00	Least significant 2 bits of the output port 1 value if the extended route table mechanism is enabled. Most significant 2 bits of the route value if bit 19 of the Processing Element Features CAR is set. This field shall be reserved if extended route table mechanism is not enabled and bit 19 of the Processing Element Features CAR is clear.
24-31	Config_output_port	see footnote[1]	Configuration output port. If bit 19 of the Processing Element Features CAR is set, the routing table value read and written is found in the Config_output_port1_lsb and Config_output_port fields.

[1]The Config_output_port reset value is implementation dependent

3.5.7 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 3-11.

Table 3-11. Bit Settings for Standard Route Default Port CSR

Bits	Name	Reset Value	Description
0–3	Implementation Specific	0x0	These bits optionally control implementation-specific routing functionality. This field is allowed when bit 19 of the Processing Element Features CAR is set. If bit 19 of the Processing Element Features CAR is clear, this field is reserved. Bits in this field that do not control implementation specific routing functionality shall be read only, with a fixed value of 0. Implementation-specific routing functionality may be active if any bit in this field is set. Implementation-specific routing functionality shall not be active if all bits in this field are clear.
4–21	—		Reserved
22-23	Route Type	0b11	Extended value for packet routing. This field is required when bit 19 of the Processing Element Features CAR is set. If bit 19 of the Processing Element Features CAR is clear, this field is reserved.
24–31	Default_output_port	0x00	Default output port When bit 19 of the Processing Element Features CAR is set, Route Type concatenated with default_output_port shall be encoded as follows: 0x000 to 0x0FF - Egress Port Number 0x00 to 0xFF 0x100 to 0x1FF - Multicast Mask Number 0x00 to 0xFF 0x200 to 0x2FF - Reserved. 0x300 - Drop Packet 0x301 to 0x3FF - Reserved. Selection of an Egress Port Number which is not supported by the device, or a Multicast Mask Number which is not supported by the device, shall result in implementation specific routing behavior.

3.6 Switch Routing Table Register Block

A switch device which has bit 19 set in the Processing Element Features CAR shall implement this register block.

3.6.1 Register Map

The register map for the routing table registers shall be as specified by Table 3-12. This register map is currently only defined for devices with up to 16 RapidIO ports, but can be extended or shortened if more or less port definitions are required for a device. For example, a device with four RapidIO ports is only required to use register map space corresponding to offsets [EF_PTR+0x00] through [EF_PTR+0xBC]. Register map offset [EF_PTR+0x140] can be used for another Extended Features block.

Table 3-12. Switch Routing Table Register Map

	Block Byte Offset	Register Name
General	0x0	Routing Table Register Block Header
	0x4-0x1C	Reserved
Broadcast	0x20	Broadcast Routing Table Control CSR
	0x24-0x2C	Reserved
	0x30	Broadcast Level 0 Info CSR
	0x34	Broadcast Level 1 Info CSR
	0x38	Broadcast Level 2 Info CSR
	0x3C	Reserved
Port 0	0x40	Port 0 Routing Table Control CSR
	0x44-0x4C	Reserved
	0x50	Port 0 Level 0 Info CSR
	0x54	Port 0 Level 1 Info CSR
	0x58	Port 0 Level 2 Info CSR
	0x5C	Reserved
Port 1	0x60	Port 1 Routing Table Control CSR
	0x64-0x6C	Reserved
	0x70	Port 1 Level 0 Info CSR
	0x74	Port 1 Level 1 Info CSR
	0x78	Port 1 Level 2 Info CSR
	0x7C	Reserved

Table 3-12. Switch Routing Table Register Map

	Block Byte Offset	Register Name
Ports 2-14	0x80–21C	Assigned to Port 2-14 CSRs
Port 15	0x220	Port 15 Routing Table Control CSR
	0x224-0x22C	Reserved
	0x230	Port 15 Level 0 Info CSR
	0x234	Port 15 Level 1 Info CSR
	0x238	Port 15 Level 2 Info CSR
	0x23C	Reserved

3.6.2 Switch Routing Table Register Block Header (Block Offset 0x0)

The switch routing table register block header register contains the EF_PTR to the next EF_BLK and the EF_ID that identifies this as the switch routing table registers block header. The use and meaning of the bits and bit fields of this register shall be as specified in Table 3-13. The register shall be read-only.

Table 3-13. Bit Settings for Switch Routing Table Register Block Header

Bit	Name	Reset Value	Description
0-15	EF_PTR	see footnote[1]	Hard wired pointer to the next block in the data structure, if one exists
16-31	EF_ID	0x000E	Hard wired Extended Features ID

[1]The EF_PTR reset value is implementation dependent

3.6.3 Broadcast Routing Table Control CSR (Block Offset 0x20)

Writes to this register are broadcast to all Port n Routing Table Control CSRs. The use and meaning of the bits and bit fields of this register shall be as specified in Table 3-14. Unless otherwise specified, the bits and bit fields in this register are write only.

Table 3-14. Bit Settings for Port n Routing Table Control CSR

Bit	Name	Reset Value	Description
0	Three Levels	0b1	0 - Routing table entries support a contiguous range of device IDs, starting with device ID 0x00/0x0000/0x00000000. 1 - Routing table entries support a hierarchical routing scheme
1	Dev32 Route Control	0b0	0 - Dev32 Device IDs are routed using Byte 0 for Level 0, Byte 1 for Level 1, and Byte 2 for Level 2 1 - Dev32 Device IDs are routed using Byte 1 for Level 0, Byte 2 for Level 1, and Byte 3 for level 2 Reserved if Three Levels is clear.
2-31	___		Reserved

3.6.4 Broadcast Level 0 Info CSR (Block Offset 0x30)

This register shall communicate the location of the Broadcast Level 0 routing table group. Writes to the Broadcast Level 0 routing table group affect all Port n Level 0 routing groups. The use and meaning of the bits and bit fields of this register shall be as specified in Table 3-15. This register shall be read only.

Table 3-15. Bit Settings for Broadcast Level 0 Info CSR

Bit	Name	Reset Value	Description
0-7	Num_L0_Groups	see footnote[1]	Communicates the number of 256 entry routing table groups for Level 0. When Three Levels is 0, Num_L0_Groups shall communicate the number of groups available for routing. When Three Levels is 1, Num_L0_Groups shall be 1. Num_L0_Groups is encoded as follows: 0x00 - 256 Groups 0x01 - 1 Group 0x02 - 2 Groups 0x03 - 3 Groups ... 0xFF - 255 Groups
8-21	L0_Group_Ptr	see footnote[2]	The L0_Group_Ptr value shall be the maintenance offset of the first entry in the first routing table group for level 0, divided by 1024. The maintenance offset of the first entry in the first routing group for level 0 shall be a 1024 byte aligned address. The L0_Group_Ptr value shall indicate an address in Implementation Defined register space. Writes to the broadcast routing table group entries pointed to by this register shall cause the corresponding routing table group entries for all ports to assume the value written. Implementation specific behavior shall occur for writes to routing table group entries if the contents of the Port n Routing Table Control CSRs are not the same for all ports .
22-31	___	All 0's	Reserved

[1]The Num_L0_Groups reset value is implementation dependent

[2]The L0_Group_Ptr reset value is implementation dependent

3.6.5 Broadcast Level 1 Info CSR (Block Offset 0x34)

This register shall communicate the location of the Broadcast Level 1 routing table group. When Three Levels is 0, this register is reserved. The use and meaning of the bits and bit fields of this register shall be as specified in Table 3-16. This register shall be read only.

Table 3-16. Bit Settings for Broadcast Level 1 Info CSR

Bit	Name	Reset Value	Description
0-7	Num_L1_Groups	see footnote[1]	Communicates the number of 256 entry routing table groups for Level 1. Num_L1_Groups shall be encoded as follows: 0x00 - 256 Groups 0x01 - 1 Group 0x02 - 2 Groups 0x03 - 3 Groups ... 0xFF - 255 Groups
8-21	L1_Group_Ptr	see footnote[2]	The L1_Group_Ptr value shall be the maintenance offset of the first entry in the first routing table group for level 1, divided by 1024. The maintenance offset of the first entry in the first routing group for level 1 shall be a 1024 byte aligned address. The L1_Group_Ptr value shall indicate an address in Implementation Defined register space. Writes to the broadcast routing table group entries pointed to by this register shall cause the corresponding routing table group entries for all ports to assume the value written. Implementation specific behavior shall occur for writes to routing table group entries if the contents of the Port n Routing Table Control CSRs are not the same for all ports .
22-31	___	All 0's	Reserved

[1]The Num_L1_Groups reset value is implementation dependent

[2]The L1_Group_Ptr reset value is implementation dependent

3.6.6 Broadcast Level 2 Info CSR (Block Offset 0x38)

This register shall communicate the location of the Level 2 routing table group for Port n. When Three Levels is 0, this register is reserved. The use and meaning of the bits and bit fields of this register shall be as specified in Table 3-17. This register shall be read only.

Table 3-17. Bit Settings for Broadcast Level 2 Info CSR

Bit	Name	Reset Value	Description
0-7	Num_L2_Groups	see footnote[1]	Communicates the number of 256 entry routing table groups for Level 2. Num_L2_Groups shall be encoded as follows: 0x00 - 256 Groups 0x01 - 1 Group 0x02 - 2 Groups 0x03 - 3 Groups ... 0xFF - 255 Groups
8-21	L2_Group_Ptr	see footnote[2]	The L2_Group_Ptr value shall be the maintenance offset of the first entry in the first broadcast routing table group for level 2, divided by 1024. The maintenance offset of the first entry in the first broadcast routing group for level 2 shall be a 1024 byte aligned address. The L2_Group_Ptr value shall indicate an address in Implementation Defined register space. Writes to the broadcast routing table group entries pointed to by this register shall cause the corresponding routing table group entries for all ports to assume the value written. Implementation specific behavior shall occur for writes to broadcast routing table group entries if the contents of the Port n Routing Table Control CSRs are not the same for all ports .
22-31	___	All 0's	Reserved

[1]The Num_L2_Groups reset value is implementation dependent

[2]The L2_Group_Ptr reset value is implementation dependent

3.6.7 Port *n* Routing Table Control CSRs (Block Offset 0x40 + (0x20 * n))

These registers shall control the routing mode for all ports whose Port n Level 0 Info CSR L0_Group_Ptr field value is the same. The use and meaning of the bits and bit fields of these registers shall be as specified in Table 3-18. Unless otherwise specified, the bits and bit fields in these registers are read/write.

Table 3-18. Bit Settings for Port *n* Routing Table Control CSRs

Bit	Name	Reset Value	Description
0	Three Levels	0b1	0 - Routing table entries support a contiguous range of device IDs, starting with device ID 0x00/0x0000/0x00000000. 1 - Routing table entries support a hierarchical routing scheme
1	Dev32 Route Control	0b0	0 - Dev32 Device IDs are routed using Byte 0 for Level 0, Byte 1 for Level 1, and Byte 2 for Level 2 1 - Dev32 Device IDs are routed using Byte 1 for Level 0, Byte 2 for Level 1, and Byte 3 for level 2 Reserved if Three Levels is clear.
2-31	___		Reserved

3.6.8 Port *n* Level 0 Info CSRs (Block Offset 0x50 + (0x20 * n))

These registers shall communicate the location of the Level 0 routing table group for Port n. The use and meaning of the bits and bit fields of these registers shall be as specified in Table 3-19. These registers shall be read only.

Table 3-19. Bit Settings for Port *n* Level 0 Info CSRs

Bit	Name	Reset Value	Description
0-7	Num_L0_Groups	see footnote[1]	Communicates the number of 256 entry routing table groups for Level 0. When Three Levels is 0, Num_L0_Groups shall communicate the number of groups available for routing. When Three Levels is 1, Num_L0_Groups shall be 1. Num_L0_Groups is encoded as follows: 0x00 - 256 Groups 0x01 - 1 Group 0x02 - 2 Groups 0x03 - 3 Groups ... 0xFF - 255 Groups
8-21	L0_Group_Ptr	see footnote[2]	The L0_Group_Ptr value shall be the maintenance offset of the first entry in the first routing table group for level 0, divided by 1024. The maintenance offset of the first entry in the first routing group for level 0 shall be a 1024 byte aligned address. The L0_Group_Ptr value shall indicate an address in Implementation Defined register space. All ports with identical L0_Group_Ptr values shall have identical Level 0 routing behavior.
22-31	___	All 0's	Reserved

[1]The Num_L0_Groups reset value is implementation dependent

[2]The L0_Group_Ptr reset value is implementation dependent

3.6.9 Port *n* Level 1 Info CSRs (Block Offset 0x54 + (0x20 * n))

These registers shall communicate the location of the Level 1 routing table group for Port n. When the Three Levels of the Port n Routing Table Control CSRs is 0, these registers shall be reserved. The use and meaning of the bits and bit fields of these registers shall be as specified in Table 3-20. These registers shall be read only.

Table 3-20. Bit Settings for Port *n* Level 1 Info CSRs

Bit	Name	Reset Value	Description
0-7	Num_L1_Groups	see footnote[1]	Communicates the number of 256 entry routing table groups for Level 1. Num_L1_Groups shall be encoded as follows: 0x00 - 256 Groups 0x01 - 1 Group 0x02 - 2 Groups 0x03 - 3 Groups ... 0xFF - 255 Groups
8-21	L1_Group_Ptr	see footnote[2]	The L0_Group_Ptr value shall be the maintenance offset of the first entry in the first routing table group for level 0, divided by 1024. The maintenance offset of the first entry in the first routing group for level 0 shall be a 1024 byte aligned address. The L1_Group_Ptr value shall indicate an address in Implementation Defined register space. All ports with identical L1_Group_Ptr values shall have identical Level 1 packet routing behavior.
22-31	___	All 0's	Reserved

[1]The Num_L1_Groups reset value is implementation dependent

[2]The L1_Group_Ptr reset value is implementation dependent

3.6.10 Port *n* Level 2 Info CSRs (Block Offset 0x58 + (0x20 * n))

These registers shall communicate the location of the Level 2 routing table group for Port n. When the Three Levels of the Port n Routing Table Control CSRs is 0, these registers shall be reserved. The use and meaning of the bits and bit fields of these registers shall be as specified in Table 3-21. These registers shall be read only.

Table 3-21. Bit Settings for Port *n* Level 2 Info CSRs

Bit	Name	Reset Value	Description
0-7	Num_L2_Groups	see footnote[1]	Communicates the number of 256 entry routing table groups for Level 2. Num_L2_Groups shall be encoded as follows: 0x00 - 256 Groups 0x01 - 1 Group 0x02 - 2 Groups 0x03 - 3 Groups ... 0xFF - 255 Groups
8-21	L2_Group_Ptr	see footnote[2]	The L2_Group_Ptr value shall be the maintenance offset of the first entry in the first routing table group for level 2, divided by 1024. The maintenance offset of the first entry in the first routing group for level 2 shall be a 1024 byte aligned address. The L1_Group_Ptr value shall indicate an address in Implementation Defined register space. All ports with identical L2_Group_Ptr values shall have identical Level 2 routing behavior.
22-31	___	All 0's	Reserved

[1]The Num_L2_Groups reset value is implementation dependent

[2]The L2_Group_Ptr reset value is implementation dependent

3.7 Routing Table Group Register Format

A group of routing table entries consists of 256 consecutive registers. The L0_Group_Ptr, L1_Group_Ptr, and L2_Group_Ptr point to the first entry of the first group of a number of contiguous groups of register entries.

The address of registers in a routing table group is computed using three values, denoted as Group_Ptr, X, and Y, where:

- Group_Ptr is the value of the "Lz_Group_Ptr" field found in the Port n Level z Info CSRs
- X is the group number
- Y is the entry number within the group

As shown in the following register format definitions, the register address is computed as:

(Group_Ptr * 0x400) + (X * 0x400) + (Y * 4).

For example, assume that the Port n Level z CSR value is 0x03048C00 and it is necessary to address entry number 127 in group number 3. The address computation is:

(0x123 * 0x400) + (3 * 0x400) + (127 * 4) = 0x499FC

3.7.1 Broadcast Level 0 Group x Entry y Routing Table Entry CSR (Offset = (L0_Group_Ptr*0x400) + (x * 0x400) + (y*4))

Writes to the Broadcast Level 0 Group x Entry y Routing Table Entry CSRs shall cause the corresponding Port n Level 0 Group x Entry y Routing Table Entry CSRs for all ports to assume the value written. The use and meaning of the bits and bit fields of this register shall be as specified in Table 3-22. The bits and bit fields in this register are write only.

Table 3-22. Bit Settings for Broadcast Level 0 Group x Entry y Routing Table Entry CSR

Bit	Name	Reset Value	Description
0-3	Implementation-defined	Impl. Spec	These bits control implementation specific behavior. When these bits are written to 0x0, no implementation specific function shall be invoked.
4-21	___		Reserved
22-31	Routing Value	0x300	Routing table entry 0x000 to 0x0FF - Egress Port Number 0x00 to 0xFF 0x100 to 0x1FF - Multicast Mask Number 0x00 to 0xFF. Refer to Part 11. 0x200 to 0x2FF - Level 1 Group number 0x00 to 0xFF 0x300 - Drop Packet 0x301 - Use value found in Standard Port Default Route CSR 0x302 to 0x3FF - Reserved. Selection of an Egress Port Number, Multicast Mask or Level 1 Group Number which does not exist in the device shall result in implementation specific behavior. When the Three Levels field of the Port n Routing Table Control CSR is clear, the values 0x200 through 0x2FF shall result in implementation specific routing behavior.

3.7.2 Broadcast Level 1 Group x Entry y Routing Table Entry CSR (Offset = (L1_Group_Ptr*0x400) + (x * 0x400) + (y*4))

Writes to the Broadcast Level 1 Group x Entry y Routing Table Entry CSRs shall cause the corresponding Port n Level 1 Group x Entry y Routing Table Entry CSRs for all ports to assume the value written. The use and meaning of the bits and bit fields of this register shall be as specified in Table 3-23. The bits and bit fields in this register are write only.

Table 3-23. Level 1 Group x Entry y Routing Table Entry CSR

Bit	Name	Reset Value	Description
0-3	Implementation-defined	Impl. Spec	These bits control implementation specific behavior. When these bits are written to 0x0, no implementation specific function shall be invoked.
4-21	___		Reserved
22-31	Routing Value	0x300	Routing table entry 0x000 to 0x0FF - Egress Port Number 0x00 to 0xFF 0x100 to 0x1FF - Multicast Mask Number 0x00 to 0xFF. Refer to Part 11. 0x200 to 0x2FF - Level 2 Group number 0x00 to 0xFF 0x300 - Drop Packet 0x301 - Use value found in Standard Port Default Route CSR 0x302 to 0x3FF - Reserved. Selection of an Egress Port Number, Multicast Mask or Level 2 Group Number which does not exist in the device shall result in implementation specific behavior.

3.7.3 Broadcast Level 2 Group x Entry y Routing Table Entry CSR

(Offset = (L2_Group_Ptr*0x400) + (x * 0x400) + (y*4))

Writes to the Broadcast Level 2 Group x Entry y Routing Table Entry CSRs shall cause the corresponding Port n Level 2 Group x Entry y Routing Table Entry CSRs for all ports to assume the value written. The use and meaning of the bits and bit fields of this register shall be as specified in Table 3-24. The bits and bit fields in this register are write only.

Table 3-24. Bit Settings for Broadcast Level 2 Group x Entry y Routing Table Entry CSR

Bit	Name	Reset Value	Description
0-3	Implementation-defined	Impl. Spec	These bits control implementation specific behavior. When these bits are written to 0x0, no implementation specific function shall be invoked.
4-21	___		Reserved
22-31	Routing Value	0x300	Routing table entry 0x000 to 0x0FF - Egress Port Number 0x00 to 0xFF 0x100 to 0x1FF - Multicast Mask Number 0x00 to 0xFF. Refer to Part 11. 0x200 to 0x2FF - Reserved. 0x300 - Drop Packet 0x301 - Route packet using the Standard Port Default Route CSR 0x302 to 0x3FF - Reserved. Selection of an Egress Port Number or Multicast Mask Number which does not exist in the device shall result in implementation specific behavior.

3.7.4 Level 0 Group x Entry y Routing Table Entry CSR (Offset = (L0_Group_Ptr*0x400) + (x * 0x400) + (y*4))

This register shall control the routing mode for all ports whose Port n Level 0 Info CSR L0_Group_Ptr field value is the same. The use and meaning of the bits and bit fields of this register shall be as specified in Table 3-25. Unless otherwise specified, the bits and bit fields in this register are read/write.

Table 3-25. Bit Settings for Level 0 Group x Entry y Routing Table Entry CSR

Bit	Name	Reset Value	Description
0-3	Implementation-defined	Impl. Spec	These bits control implementation specific behavior. When these bits are written to 0x0, no implementation specific function shall be invoked.
4-21	___		Reserved
22-31	Routing Value	0x300	Routing table entry 0x000 to 0x0FF - Egress Port Number 0x00 to 0xFF 0x100 to 0x1FF - Multicast Mask Number 0x00 to 0xFF. Refer to Part 11. 0x200 to 0x2FF - Level 1 Group number 0x00 to 0xFF 0x300 - Drop Packet 0x301 - Use value found in Standard Port Default Route CSR 0x302 to 0x3FF - Reserved. Selection of an Egress Port Number, Multicast Mask or Level 1 Group Number which does not exist in the device shall result in implementation specific behavior. When the Three Levels field of the Port n Routing Table Control CSR is clear, the values 0x200 through 0x2FF shall result in implementation specific routing behavior.

3.7.5 Level 1 Group x Entry y Routing Table Entry CSR (Offset = (L1_Group_Ptr*0x400) + (x * 0x400) + (y*4))

This register shall control the routing mode for all ports whose Port n Level 1 Info CSR L1_Group_Ptr field value is the same. The use and meaning of the bits and bit fields of this register shall be as specified in Table 3-26. Unless otherwise specified, the bits and bit fields in this register are read/write.

Table 3-26. Bit Settings for Level 1 Group x Entry y Routing Table Entry CSR

Bit	Name	Reset Value	Description
0-3	Implementation-defined	Impl. Spec	These bits control implementation specific behavior. When these bits are written to 0x0, no implementation specific function shall be invoked.
4-21	___		Reserved
22-31	Routing Value	0x300	Routing table entry 0x000 to 0x0FF - Egress Port Number 0x00 to 0xFF 0x100 to 0x1FF - Multicast Mask Number 0x00 to 0xFF. Refer to Part 11. 0x200 to 0x2FF - Level 2 Group number 0x00 to 0xFF 0x300 - Drop Packet 0x301 - Use value found in Standard Port Default Route CSR 0x302 to 0x3FF - Reserved. Selection of an Egress Port Number, Multicast Mask or Level 2 Group Number which does not exist in the device shall result in implementation specific behavior.

3.7.6 Level 2 Group x Entry y Routing Table Entry CSR (Offset = (L2_Group_Ptr*0x400) + (x * 0x400) + (y*4))

This register shall control the routing mode for all ports whose Port n Level 2 Info CSR L2_Group_Ptr field value is the same. The use and meaning of the bits and bit fields of this register shall be as specified in Table 3-27. Unless otherwise specified, the bits and bit fields in this register are read/write.

Table 3-27. Bit Settings for Level 2 Group x Entry y Routing Table Entry CSR

Bit	Name	Reset Value	Description
0-3	Implementation-defined	Impl. Spec	These bits control implementation specific behavior. When these bits are written to 0x0, no implementation specific function shall be invoked.
4-21	___		Reserved
22-31	Routing Value	0x300	Routing table entry 0x000 to 0x0FF - Egress Port Number 0x00 to 0xFF 0x100 to 0x1FF - Multicast Mask Number 0x00 to 0xFF. Refer to Part 11. 0x200 to 0x2FF - Reserved. 0x300 - Drop Packet 0x301 - Route packet using the Standard Port Default Route CSR 0x302 to 0x3FF - Reserved. Selection of an Egress Port Number or Multicast Mask Number which does not exist in the device shall result in implementation specific behavior.

Blank page

Annex A Dev32 Hierarchical Programming Model (Informative)

A.1 Dev32 Configuration Examples

This chapter provides several examples of how to use the Dev32 routing table programming interface. The given examples build upon each other while proceeding through the sections. References to the order of operations within the examples run from the top of a list to the bottom unless otherwise stated.

Initially assume a switch with 16 ports which supports Dev32 device IDs. Assume that the switch must support the following routing hierarchy, where "**" means "All Values":

- Device ID 0x00_11_20_** must be routed to port 14.
- Device IDs 0x00_11_0X_** must be routed to port X, where X is 0 to 13.
- Device IDs 0x00_ZZ_**_** must be routed to port 15 when ZZ is 0 to 0x10.
- All other packets must be dropped.

Further assume that Port 7 must be programmed to support the above hierarchy, and has initial register values as follows:

Table A-1. Example Port 7 Routing Table Register Block Registers

Register Name	Register Address	Register Value
Switch Routing Table Register Block Header	0x8000	N/A
Port 7 Routing Table Control CSR	0x8120	0x8000_0000
Port 7 Level 0 Info CSR	0x8130	0x0107_0000
Port 7 Level 1 Info CSR	0x8134	0x0307_0400
Port 7 Level 2 Info CSR	0x8138	0x0407_1000

A.1.1 Example 1: Routing 0x00_11_20_** to Port 14

To route the Dev32 destination IDs 0x00_11_20_** to Port 14, make use of routing table group 0 for Level 0, and routing table group 1 for Level 1 and Level 2. Specific entries for each level must

be programmed.

Table A-2. Example 1 Accesses

Register Name	Register Address	Register Value	Description
Port 7 Level 0 Group 0 Entry 0 Routing Table Entry CSR	0x0007_0000	0x0000_0201	Map Level 0 Group 0 index 0x00 to Level 1 Group 1
Port 7 Level 1 Group 1 Entry 0x11 Routing Table Entry CSR	0x0007_0844	0x0000_0201	Map Level 1 Group 1 index 0x11 to Level 2 Group 1
Port 7 Level 2 Group 1 Entry 0x20 Routing Table Entry CSR	0x0007_1480	0x0000_000E	Map Level 2 Group 1 index 0x20 to Port 14.

A.1.2 Example 2: Routing 0x00_11_0X_** to Port X

This example builds upon the configuration put in place by Example 1. Routing configuration is therefore complete for Level 0 and Level 1, so what remains is to complete the Level 2 programming.

Table A-3. Example 2 Accesses

Register Name	Register Address	Register Value	Description
Port 7 Level 2 Group 1 Entry 0x00 Routing Table Entry CSR	0x0007_1400	0x0000_0000	Map Level 2 Group 1 index 0x00 to Port 0.
Port 7 Level 2 Group 1 Entry 0x01 Routing Table Entry CSR	0x0007_1404	0x0000_0001	Map Level 2 Group 1 index 0x01 to Port 1.
Port 7 Level 2 Group 1 Entry 0x02 Routing Table Entry CSR	0x0007_1408	0x0000_0002	Map Level 2 Group 1 index 0x02 to Port 2.
Port 7 Level 2 Group 1 Entry 0x03 Routing Table Entry CSR	0x0007_140C	0x0000_0003	Map Level 2 Group 1 index 0x03 to Port 3.
Port 7 Level 2 Group 1 Entry 0x04 Routing Table Entry CSR	0x0007_1410	0x0000_0004	Map Level 2 Group 1 index 0x04 to Port 4.
Port 7 Level 2 Group 1 Entry 0x05 Routing Table Entry CSR	0x0007_1414	0x0000_0005	Map Level 2 Group 1 index 0x05 to Port 5.
Port 7 Level 2 Group 1 Entry 0x06 Routing Table Entry CSR	0x0007_1418	0x0000_0006	Map Level 2 Group 1 index 0x06 to Port 6.
Port 7 Level 2 Group 1 Entry 0x07 Routing Table Entry CSR	0x0007_141C	0x0000_0007	Map Level 2 Group 1 index 0x07 to Port 7.
Port 7 Level 2 Group 1 Entry 0x08 Routing Table Entry CSR	0x0007_1420	0x0000_0008	Map Level 2 Group 1 index 0x08 to Port 8.
Port 7 Level 2 Group 1 Entry 0x09 Routing Table Entry CSR	0x0007_1424	0x0000_0009	Map Level 2 Group 1 index 0x09 to Port 9.
Port 7 Level 2 Group 1 Entry 0x0A Routing Table Entry CSR	0x0007_1428	0x0000_000A	Map Level 2 Group 1 index 0x0A to Port A.
Port 7 Level 2 Group 1 Entry 0x0B Routing Table Entry CSR	0x0007_142C	0x0000_000B	Map Level 2 Group 1 index 0x0B to Port B.

Table A-3. Example 2 Accesses

Register Name	Register Address	Register Value	Description
Port 7 Level 2 Group 1 Entry 0x0C Routing Table Entry CSR	0x0007_1430	0x0000_000C	Map Level 2 Group 1 index 0x0C to Port C.
Port 7 Level 2 Group 1 Entry 0x0D Routing Table Entry CSR	0x0007_1434	0x0000_000D	Map Level 2 Group 1 index 0x0D to Port D.

A.1.3 Example 3: Routing 0x00_ZZ_**_** to Port 15, ZZ=[0,0x10]

This example builds upon the configuration put in place by Example 1. Routing configuration is therefore complete for Level 0, so what remains is to program the Level 1 registers.

Table A-4. Example 3 Accesses

Register Name	Register Address	Register Value	Description
Port 7 Level 1 Group 1 Entry 0x00 Routing Table Entry CSR	0x0007_0800	0x0000_000F	Map Level 1 Group 1 index 0x00 to Port 15.
Port 7 Level 1 Group 1 Entry 0x01 Routing Table Entry CSR	0x0007_0804	0x0000_000F	Map Level 1 Group 1 index 0x01 to Port 15.
Port 7 Level 1 Group 1 Entry 0x02 Routing Table Entry CSR	0x0007_0808	0x0000_000F	Map Level 1 Group 1 index 0x02 to Port 15.
Port 7 Level 1 Group 1 Entry 0x03 Routing Table Entry CSR	0x0007_080C	0x0000_000F	Map Level 1 Group 1 index 0x03 to Port 15.
Port 7 Level 1 Group 1 Entry 0x04 Routing Table Entry CSR	0x0007_0810	0x0000_000F	Map Level 1 Group 1 index 0x04 to Port 15.
Port 7 Level 1 Group 1 Entry 0x05 Routing Table Entry CSR	0x0007_0814	0x0000_000F	Map Level 1 Group 1 index 0x05 to Port 15.
Port 7 Level 1 Group 1 Entry 0x06 Routing Table Entry CSR	0x0007_0818	0x0000_000F	Map Level 1 Group 1 index 0x06 to Port 15.
Port 7 Level 1 Group 1 Entry 0x07 Routing Table Entry CSR	0x0007_081C	0x0000_000F	Map Level 1 Group 1 index 0x07 to Port 15.
Port 7 Level 1 Group 1 Entry 0x08 Routing Table Entry CSR	0x0007_0820	0x0000_000F	Map Level 1 Group 1 index 0x08 to Port 15.
Port 7 Level 1 Group 1 Entry 0x09 Routing Table Entry CSR	0x0007_0824	0x0000_000F	Map Level 1 Group 1 index 0x09 to Port 15.
Port 7 Level 1 Group 1 Entry 0x0A Routing Table Entry CSR	0x0007_0828	0x0000_000F	Map Level 1 Group 1 index 0x0A to Port 15.
Port 7 Level 1 Group 1 Entry 0x0B Routing Table Entry CSR	0x0007_082C	0x0000_000F	Map Level 1 Group 1 index 0x0B to Port 15.
Port 7 Level 1 Group 1 Entry 0x0C Routing Table Entry CSR	0x0007_0830	0x0000_000F	Map Level 1 Group 1 index 0x0C to Port 15.
Port 7 Level 1 Group 1 Entry 0x0D Routing Table Entry CSR	0x0007_0834	0x0000_000F	Map Level 1 Group 1 index 0x0D to Port 15.
Port 7 Level 1 Group 1 Entry 0x0E Routing Table Entry CSR	0x0007_0838	0x0000_000F	Map Level 1 Group 1 index 0x0E to Port 15.

Table A-4. Example 3 Accesses

Register Name	Register Address	Register Value	Description
Port 7 Level 1 Group 1 Entry 0x0F Routing Table Entry CSR	0x0007_083C	0x0000_000F	Map Level 1 Group 1 index 0x0F to Port 15.
Port 7 Level 1 Group 1 Entry 0x10 Routing Table Entry CSR	0x0007_0840	0x0000_000F	Map Level 1 Group 1 index 0x10 to Port 15.

A.1.4 Example 4: All Other Packets Must Be Dropped

This example builds upon the configuration put in place by Example 3. Routing for Dev32 device IDs has been configured. Dev16 and Dev8 deviceIDs are routed using Group 0 of Level 1 and Level 2, respectively. The default values for all entries is to drop packets. Nothing more needs to be programmed to drop all Dev16 and Dev8 deviceIDs.

A.1.5 Example 5: Flat Routing Table Operation

This example illustrates the “flat” programming model, in which device IDs are supported sequentially by the routing tables. Dev16 device IDs of the form 0x00** are treated as Dev8 device IDs.

Initially assume a switch with 16 ports which supports Dev32 device IDs. Assume that the switch must support the following routing hierarchy, where “**” means “All Values”:

- Device ID 0x01_20 must be routed to port 14.
- Device IDs 0x00_0X must be routed to port X, where X is 0 to 13.
- Device IDs 0x02_00 and 0x03_00 must be routed to port 15.
- All other packets must be dropped.

Further assume that Port 7 must be programmed to support the above hierarchy, and has initial register values as follows:

Table A-5. Example 5 Port 7 Routing Table Register Block Registers

Register Name	Register Address	Register Value
Switch Routing Table Register Block Header	0x8000	N/A
Port 7 Routing Table Control CSR	0x8120	0x8000_0000
Port 7 Level 0 Info CSR	0x8130	0x0107_0000
Port 7 Level 1 Info CSR	0x8134	0x0307_0400
Port 7 Level 2 Info CSR	0x8138	0x0407_1000

The following register accesses must be performed:

Table A-6. Example 5 Accesses

Register Name	Register Address	Register Value	Description
Port 7 Routing Table Control CSR	0x0000_8120	0x0000_0000	Change to Flat Routing Table Model
Port 7 Level 0 Info CSR	0x0000_8130	0x0407_0000	Read Level 0 Info to determine how many DeviceIDs are supported. Four groups are supported, or destIDs 0x0000 through 0x03FF.
Port 7 Level 1 Info CSR	0x0000_8134	0x0000_0000	Read Level 1 Info, confirm register is reserved
Port 7 Level 2 Info CSR	0x0000_8138	0x0000_0000	Read Level 2 Info, confirm register is reserved
Port 7 Level 0 Group 1 Entry 0x20	0x0007_0480	0x0000_000E	Route DestID 0x0120 to port 14.
Port 7 Level 0 Group 0 Entry 0x00	0x0007_0000	0x0000_0000	Route DestID 0x0000 to port 0.
Port 7 Level 0 Group 0 Entry 0x01	0x0007_0004	0x0000_0001	Route DestID 0x0001 to port 1.
Port 7 Level 0 Group 0 Entry 0x02	0x0007_0008	0x0000_0002	Route DestID 0x0002 to port 2.
Port 7 Level 0 Group 0 Entry 0x03	0x0007_000C	0x0000_0003	Route DestID 0x0003 to port 3.
Port 7 Level 0 Group 0 Entry 0x04	0x0007_0010	0x0000_0004	Route DestID 0x0004 to port 4.
Port 7 Level 0 Group 0 Entry 0x05	0x0007_0014	0x0000_0005	Route DestID 0x0005 to port 5.
Port 7 Level 0 Group 0 Entry 0x06	0x0007_0018	0x0000_0006	Route DestID 0x0006 to port 6.
Port 7 Level 0 Group 0 Entry 0x07	0x0007_001C	0x0000_0007	Route DestID 0x0007 to port 7.
Port 7 Level 0 Group 0 Entry 0x08	0x0007_0020	0x0000_0008	Route DestID 0x0008 to port 8.
Port 7 Level 0 Group 0 Entry 0x09	0x0007_0024	0x0000_0009	Route DestID 0x0009 to port 9.
Port 7 Level 0 Group 0 Entry 0x0A	0x0007_0028	0x0000_000A	Route DestID 0x000A to port 10.
Port 7 Level 0 Group 0 Entry 0x0B	0x0007_002C	0x0000_000B	Route DestID 0x000B to port 11.
Port 7 Level 0 Group 0 Entry 0x0C	0x0007_0030	0x0000_000C	Route DestID 0x000C to port 12.
Port 7 Level 0 Group 0 Entry 0x0D	0x0007_0034	0x0000_000D	Route DestID 0x000D to port 13.
Port 7 Level 0 Group 2 Entry 0x00	0x0007_0800	0x0000_000F	Route DestID 0x0200 to port 15.
Port 7 Level 0 Group 3 Entry 0x00	0x0007_0C00	0x0000_000F	Route DestID 0x0300 to port 15.

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.